EVALAI: EVALUATING AI SYSTEMS AT SCALE

A Thesis
Presented to
The Academic Faculty

By

Deshraj

In Partial Fulfillment
of the Requirements for the Degree
Master in Science in the
School of Interactive Computing

Georgia Institute of Technology

December 2018

EVALAI: EVALUATING AI SYSTEMS AT SCALE

Approved by:

Dr. Dhruv Batra, Advisor
School of Interactive Computing
Georgia Institute of Technology

Dr. Devi Parikh
School of Interactive Computing
Georgia Institute of Technology

Dr. Stefan Lee
School of Interactive Computing
Georgia Institute of Technology

Date Approved: November 20, 2018

Be not afraid of anything. You will do marvellous work. The moment you fear, you are nobody. It is fear that is the great cause of misery in the world. It is fear that is the cause of all our woes and it is fearlessness that brings heaven even in a moment.

Swami Vivekananda

To my parents, my elder brother, and my maternal grandfather for their endless love, care, support, and guidance.

To *Dr. Dhruv Batra* and *Dr. Devi Parikh* for being the awesome advisors.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Dr. Dhruv Batra for giving me this opportunity to work in the Machine Learning and Perception Lab, Virginia Tech and later advising me at Georgia Tech during my graduate school. He consistently allowed this thesis to be my own work but steered me in the right the direction whenever he thought I needed it. I am really glad that I had the opportunity to work with Dr. Devi Parikh on multiple projects such as Vicki Demo, Visual Dialog, GuessWhich, Theory of AI's mind, and EvalAI etc. I have learned a lot from Devi on how to manage your time and follow your calendar. I am always amazed by her skill of completing all the to-dos on time and prepare for the next few weeks in advance. I am also grateful to Dr. Stefan Lee for all the constant motivation and guidance he provided me throughout my internship and my graduate school at Georgia Tech. I have enjoyed interacting with him both professionally and outside of work. His valuable feedback on my rusty job talk presentation helped me to secure the job at one of the best Tech Companies in the USA. He is also a really fun person and always ready to play board games. Working with Harsh on both CloudCV and Snap Internship Project was a lot of fun. He is a great mentor and always helped me to take the right decision in my career. I truly feel privileged to have worked with him. I would also like to thank our EvalAI team for trusting me and working with me in building EvalAI. Working with Taranjeet, Akash, Shiv, Prithvi, Rishabh, and Shivani was an amazing experience.

I was really fortunate to work with some amazing students and mentors during the Google Summer of Code and Google Code-In. Working with Karan, Ayush, Utsav, Utkarsh, Ram, Deepesh, Vipin, Adarsh was a lot of fun during Google Summer of Code where we built a lot of things that will hopefully benefit the community. Their hard work, dedication and willingness to solve difficult problems inspired us to think big and aim higher.

I am really fortunate to be a member of Visual Intelligence, Machine Learning and Perception Lab. The lab is full of really hard-working, smart people who are always ready to

work on interesting and challenging research directions. I am really glad that I have direct access to these people. I have had the pleasure to work with Prithvi on multiple projects and the way he functions is just shockingly amazing. I have learned a lot after working with him. His knowledge about mathematics and his curiosity to learn new things are really commendable. I was also fortunate to team up with Arjun on multiple projects. The discussions with Arjun on different projects, daily schedule, life, work-life balance, etc have been really helpful. I would also like to thank Ram for cooking delicious food for us at both Virginia Tech and at Georgia Tech. He is one of the simplest people I have ever seen in my life. I am really thankful to Rama, Harsh, and Mohit for making the final semester at Georgia Tech a lot of fun with FIFA to release the pressure of exams, job research, and research etc.

It has been said, A friend is someone who gives you total freedom to be yourself. I am really glad to have friends like Shubhi, Abhishek, Shivam, Shiv, Aakarshika, Harsh, Ayush, Aakash, Vipul, Shubham, and Nawaz in my life who have filled my life with a lot of fun and let me be who I am.

Finally, I must express my very profound gratitude to my parents and to my brother for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Artificial Intelligence research has progressed tremendously in the last few years. There has been the introduction of several new multimodal datasets and tasks and due to which it is becoming much harder to compare new algorithms with existing ones. To solve this problem, this thesis introduces EvalAI, an open source platform for evaluating and comparing machine learning (ML) and artificial intelligence algorithms (AI) at scale. This platform is built to provide an open source, standardized, scalable solution for evaluating learned models using automatic metrics as well as with human-in-the-loop evaluation. This aims to help researchers, students, and data scientists to create, collaborate, and participate in artificial intelligence challenges organized around the globe. By simplifying and standardizing the process of benchmarking these models, EvalAI seeks to lower the barrier to entry for participating in the global scientific effort to push the frontiers of machine learning and artificial intelligence, increasing the rate of measurable progress in these communities.

CHAPTER 1
INTRODUCTION

Time and again across different scientific and engineering fields, the formulation and creation of the right question, task, and dataset to study a problem has coalesced fields around particular challenges – driving scientific progress. Likewise, progress on important problems in the fields of Computer Vision (CV) and Artificial Intelligence (AI) has been driven by the introduction of bold new tasks together with the curation of large, realistic datasets [1, 2]. Not only do these tasks and datasets establish new problems and provide data necessary to address them, but importantly they also establish reliable benchmarks where proposed solutions and hypothesis can be tested, an essential part of the scientific process. In recent years, the development of centralized evaluation platforms have lowered the barrier to compete and share results on these problems. As a result, a thriving community of data scientists and researchers has grown around these tasks, increasing the pace of progress and technical dissemination.

1.1 Challenges

1.1.1 Model evaluation in isolation

Historically, the community has focused on traditional AI tasks such as image classification, scene recognition, and sentence parsing that follow a standard *input-output* paradigm for which models can be evaluated in isolation using simple automatic metrics like accuracy, precision or recall. But with the success of deep learning techniques on a wide variety of tasks and the proliferation of 'smart' applications, there is an imminent need to evaluate AI systems in the context of human collaborators, not just in isolation.

This is especially true as AI systems become more commonplace and we find ourselves

interacting with AI agents on a daily basis. For instance, people frequently interact with virtual assistants like Alexa, Siri, or Google Assistant to get answers to their questions, to book appointments at a restaurant, or to reply to emails and messages automatically. Another such example is the use of AI for recognizing content in images, helping visually impaired users interpret the surrounding scene. To this end, the AI community has introduced several challenging high-level AI tasks - from goal oriented dialog and question answering about short articles, images or even videos to competing against the best human players in games like Go [3] or multiplayer video games like DOTA [4].

As AI improves and takes on these more difficult, high-level tasks that are poorly described by an *input-output* paradigm, there are a number of challenges to robust evaluation. Generating a natural language description for an image, having a conversation with a human, or generating aesthetically pleasing images cannot be evaluated accurately using automatic metrics – *What makes a description good or bad? Or a conversation? By what measure is an image aesthetically pleasing?*. Instead, these tasks require human evaluation; however, existing evaluation platforms do not support tasks that require human-in-the-loop evaluation. A key technical challenge to human-in-the-loop evaluation is to connect evaluation servers with a human workforce such as Amazon Mechanical Turk (AMT) [5].

1.1.2 Evaluation inside environments

Furthermore, the rise of reinforcement learning based problems in which an agent must interact with an environments introduces additional challenges for benchmarking. In contrast to supervised learning setting where performance is measured by performance on a static test set, it's less straightforward to measure generalization performance of these agents that perform sequences of actions that alter the testing environment. Evaluating these agents involves running the users code on a collection of unseen environments that constitutes a hidden test set such that one can check if algorithms "overfit" on training environments.

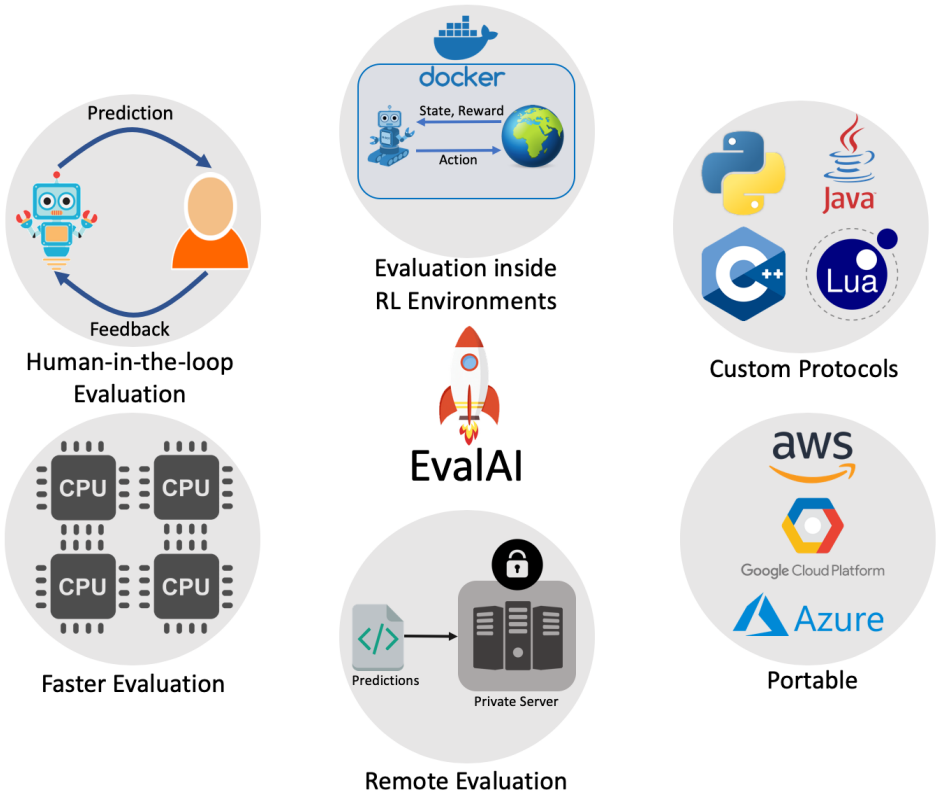


Figure 1.1: EvalAI is a new evaluation platform with the overarching goal of providing the right tools, infrastructure and framework to setup exhaustive evaluation protocols for both traditional static evaluation tasks as well as those in dynamic environments hosting multiple agents and/or humans.

1.2 System Overview

To address the aforementioned problems in the previous section, we introduce a new evaluation system for AI tasks called EvalAI. It is a highly extensible open-source platform that fulfills the critical need in the AI community for human-in-the-loop evaluation of machine learning models. The key contribution is development of a highly customizable infrastructure and a set of protocols to pair AMT users in real-time with submitted models such that humans can rate the quality of their interaction.

We have also addressed the limitations of existing platforms by supporting (1) custom evaluation pipelines written in any programming language, (2) arbitrary number of challenge phases and dataset splits, (3) remote evaluation on private worker pool, and (4) the ability

to run user's code in a dynamic environment instead of simply submitting the predictions on static dataset – enabling the evaluation of interactive agents.

By providing the functionality to connect agents, environments, and human evaluators, EvalAI enables novel research directions to be explored quickly and at scale. One such work by [6] connected human users with AI agents trained to answer questions about images in a 20-questions style image guessing game and then measured the performance of the human-AI team. By being able to run these experiments, the authors found the surprising result that performance gained through AI-AI self-play in this setting doesn't seem to generalize to human-AI teams. These sort of useful insights into the true capabilities of machine learning models once they come into contact with human users are increasingly important as more and more of these models reach consumers.

1.3 Desiderata

Having outlined the need for an evaluation platform that can properly benchmark increasingly complex tasks being tackled by the community, we explicitly specify the following 'requirements' that a modern evaluation tool should satisfy.

1.3.1 Human evaluation of agents

As discussed in the previous section, the AI community has introduced increasingly bold tasks such as goal-oriented dialog, question-answering, GuessWhich, image generation, etc. Many of these tasks require human evaluation to accurately benchmark approaches against each other. A modern evaluation platform should provide a unified framework to benchmark scenarios in which agents are not acting in isolation, but rather interacting with other agents or humans.

1.3.2 Environments, not datasets

As the community becomes more ambitious in the problems they are trying to solve, we have noticed a shift from static datasets to dynamic environments. Now instead of evaluating a model on a single task, agents are deployed in new unseen environments inside a simulation to check their generalization in novel, unseen scenarios. For instance, [7, 8]. As such, modern evaluation platforms need to be capable of running submitted agents within these environments – a significant departure from the standard evaluation paradigm of computing automatic metrics on a set of submitted predictions.

1.3.3 Extensibility

Different task require different evaluation protocols. An evaluation platform needs to support an arbitrary number of phases and dataset splits to cater to the community which often use multiple dataset splits, each serving a different purpose. For instance, MS-COCO Challenge [9], VQA [2] and Visual Dialog [10] all use multiple splits such as test-dev for validation, test-std for reporting numbers in the paper and test-challenge for announcing the winners of a challenge.

1.3.4 Modularity & Portability

Sometimes data (specially user-data belonging to organizations) is important and cannot be publicly uploaded to the evaluation platform servers that reside outside the organization's infrastructure. By decoupling the leader-board from the worker nodes that actually perform the evaluation, the platform should provide the flexibility of hosting evaluation workers independently outside the platform. This will help encourage benchmarking even when the data needs to be private. Additionally, certain evaluation tasks like running an agent inside a simulation might be compute intensive and require more powerful machines with GPUs. By allowing external workers, challenge organizers have the option of providing extra compute to meet the scalability needs of a new challenge.

1.4 Contribution

The primary contribution of this thesis is EvalAI. The outline of the thesis is as follows:

- In Chapter 2, we put our work in context of related efforts in this direction. We compare EvalAI with already existing AI challenge hosting platforms such as Kaggle, CrowdAI, CodaLab etc.

- In Chapter 3, we describe in detail the key features offered by EvalAI such as custom evaluation protocols and phases, human-in-the-loop evaluation, remote evaluation, command line interface etc.

- In Chapter 4, we discuss the system architecture of EvalAI in detail. First, we talk about how different components of EvalAI are connected together. We also mention how different dockerized services can be independently scaled according to the requirement based on metrics like CPU Utilization, RAM utilization etc. Next, we describe the individual components such as Django, NodeJS, Evaluation Worker and their significance.

- In Chapter 5, we describe the lifecycle of a challenge. We start with challenge creation using a configuration file that helps the challenge organizers to customize their challenge at the granularity of evaluation protocols and phases. Then, we talk about how the submissions are evaluated in the background with the help of the queuing system that makes the process of evaluation asynchronous.

- In Chapter 6, we discuss the case studies for two different challenges hosted on EvalAI. First we talk about Visual Question Answering challenge where we compare the performance of EvalAI with CodaLab (another challenge hosting platform) in context of Visual Question Answering Challenge. Next, we describe how we prototype the human-in-the-loop evaluation for visual dialog agents. Lastly, we describe evaluation of agents inside environments - where instead of a hidden test dataset, there are hidden test

environments, so participants have to submit pretrained models and inference code, which has to be reliably executed in these environments to benchmark them.

- In Chapter 7, we talk about the impact of the project on Open Source Community and AI community. We also discuss about our past experience of participating in Google Summer of Code (GSoC) and Google Code-In (GCI) for two years (2017, 2018).

- In Chapter 8, we briefly conclude and talk about the new features planned in the next phase of EvalAI.

- In Appendix A, we present our work on evaluating visual conversation agents via cooperative human-AI games. Our human studies suggest a counter-intuitive trend – that while AI literature shows that one version outperforms the other when paired with an AI questioner bot, we find that this improvement in AI-AI performance does not translate to improved human-AI performance. This suggests a mismatch between benchmarking of AI in isolation and in the context of human-AI teams.

- In Appendix B, We argue that for these teams to be more effective, we should also be pursuing research directions to help humans understand the strengths, weaknesses, quirks, and tendencies of AI. We instantiate these ideas in the domain of Visual Question Answering (VQA), by proposing two tasks that help measure how well a human 'understands' a VQA model (we call Vicki) – Failure Prediction (FP) and Knowledge Prediction (KP). We find that lay people indeed get better at predicting Vicki's behavior using just a few 'training' examples, but surprisingly, existing popular explanation modalities do not help make its failures or responses more predictable. While previous works have typically assessed their interpretability or their role in improving human trust, our preliminary hypothesis is that these modalities may not yet help performance of human-AI teams in a goal-driven setting.

1.5 Related Publications

- Towards better evaluation systems for AI Agents.

  Deshraj Yadav, Harsh Agrawal, Rishabh Jain, Taranjeet Singh, Akash Jain, Shiv Baran Singh, Prithvijit Chattopadhyay, Stefan Lee, Dhruv Batra.

  (In Submission)

- Fabrik: An online collaborative neural network editor.

  Utsav Garg, Viraj Prabhu, Deshraj Yadav, Ram Ramrakhya, Harsh Agrawal, Dhruv Batra.

  arXiv:1810.11649

- Do explanation modalities make VQA models more predictable to a human?

  Arjun Chandrasekaran*, Viraj Prabhu*, Deshraj Yadav*, Prithvijit Chattopadhyay*, Devi Parikh.

  2018 Conference on Empirical Methods in Natural Language Processing

- Evaluating Visual Conversational Agents via Cooperative Human-AI Games.

  Prithvijit Chattopadhyay*, Deshraj Yadav *, Viraj Prabhu, Arjun Chandrasekaran, Abhishek Das, Stefan Lee, Dhruv Batra, Devi Parikh

  5th AAAI Conference on Human Computation and Crowdsourcing (HCOMP 2017).

- It Takes Two to Tango: Towards Theory of AI's Mind.

  Arjun Chandrasekaran*, Deshraj Yadav *, Prithvijit Chattopadhyay*, Viraj Prabhu*, Devi Parikh.

  2017 Chalearn Looking at People Workshop CVPR.

- Visual Dialog.

  Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, Jos M.F. Moura, Devi Parikh, Dhruv Batra.

  IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

CHAPTER 2
RELATED WORK

Next, we survey existing evaluation platforms keeping the requirements mentioned in the previous section in mind. For ease of reading, we summarize the main capability differences in Table 2.1.

2.1 Kaggle

Kaggle [11] is one of the most popular platform for hosting competitions for data-science and machine learning. It additionally allows users to share their approach with other data scientists through cloud-based workbench that works similar to IPython notebooks. Despite the popularity of Kaggle, it has several limitations. First, being a closed-source platform limits the extensibility of the platform to support complex AI tasks that require metrics other than those available through the platform. Second, it doesn't allow multiple challenge phases – a common practice in popular challenges like VQA, Visual Dialog, MS COCO Caption Challenge. Third, Kaggle doesn't allow hosting a challenge with a private test split and private workers which limits the platform's usability by organizers that cannot share the test set publicly.

2.2 Codalab

CodaLab [12] is another open-source alternative to Kaggle providing an ecosystem for conducting computational research in a more efficient, reproducible, and collaborative manner. There are two aspects of CodaLab: worksheets and competitions. First, worksheets enable users to capture complex research pipelines in a reproducible way, creating "executable papers". By archiving the code, data and the results of an experiment, the users can precisely

Table 2.1: Head-to-head comparison of capabilities between existing challenge hosting platforms and EvalAI

Features	**Topcoder**	**Kaggle**	**CrowdAI**	**CodaLab**	**EvalAI**
Custom metrics	✗	✗	✓	✓	✓
Multiple phases/splits	✗	✗	✓	✓	✓
Open Source	✗	✗	✓	✓	✓
Remote Evaluation	✗	✗	✗	✓	✓
Human Evaluation	✗	✗	✗	✗	✓
Environments	✗	✗	✓	✗	✓

capture the research pipeline in an immutable way. Additionally, it enables researcher to present these pipelines in a comprehensible way using worksheets or notebooks written in a custom markdown language. Second, CodaLab Competitions provides an evaluation platform on which one can host their own competition and measure performance through a public leaderboard. While CodaLab Competitions is very similar to EvalAI and addresses some of the limitations of Kaggle in terms of functionality, it doesn't support interactive environments and human-in-the-loop evaluation. As the community introduces more complex tasks in which evaluation requires running an agent inside a simulation or pairing an agent with a human workforce for evaluation, a highly customizable backend like ours connected with existing platforms like Amazon Mechanical Turk become extremely important.

2.3 OpenAI Retro

Reinforcement learning algorithms also need strong evaluation and good benchmarks. A variety of benchmarks have been released, such as the Arcade Learning Environment (ALE) [13], which exposed a collection of Atari 2600 games as reinforcement learning problems, and recently the RLLab benchmark for continuous control [14]. Recently introduced **OpenAI** [15] gym is a toolkit for developing and comparing reinforcement learning algorithms. It supports teaching agents a diverse range of skills ranging from walking to playing games like pong or

pinball. The gym library provides a flexible environment agents can be evaluated using any existing numerical computation library, such as TensorFlow or PyTorch. OpenAI gym has a similar underlying philosophy of encouraging easy accessibility and reproducibility by not restricting to any particular framework. Additionally, environments are versioned in a way that will ensure that results remain meaningful and reproducible as the software is updated. Alongside the software library, OpenAI Gym has a website (gym.openai.com) where one can find scoreboards for all of the environments, showcasing results submitted by users. Users are encouraged to provide links to source code and detailed instructions on how to reproduce their results.

CHAPTER 3
EVALAI: KEY FEATURES

As discussed in the previous sections, ensuring algorithms are compared fairly in a standard way is a difficult and ultimately distracting task for AI researchers who's time would be better spent elsewhere. Establishing fair comparison requires rectifying minor differences in algorithm inputs, implementation of complex evaluation metrics, and often correct usage of non-standard dataset splits. By providing a central leaderboard and consistent evaluation protocol across different submissions, we make it easier for researchers to reproduce the results reported in papers and perform reliable and accurate analysis. We also spent considerable amount of time optimizing the evaluation code that leverage multi-core capabilities which led to decrease in evaluation time by an order of magnitude on large datasets. Additionally, we develop a unified framework to test agents by pairing them with a human worker and measuring the performance on static datasets or dynamic environments. In the following sub-sections, we describe each of those key features in more detail.

3.1 Custom Evaluation Protocal and Phases

EvalAI is highly customizable since it allows creation of arbitrary number of evaluation phases and dataset splits, evaluation using any programming language, and organizing results in both public and private leader-boards. All these services are available through an intuitive web-platform and comprehensive Rest APIs.

3.2 Human-in-the-loop Evaluation

While standard computer vision tasks such as image classification [16, 17], segmentation [18, 19], detection [19, 20] etc are easy to evaluate using the automatic metrics, it is notoriously

difficult to evaluate natural language generation tasks such as image captioning [21, 22], visual dialog [10, 23] etc using automatic metrics. Developing measures which correlate well with human judgment remains an open area of research. Automatic evaluation of models for these kind of tasks is further complicated by the huge set of possibly 'correct' responses and the relatively sparse set of ground truth annotations, even in large-scale datasets.

Given these difficulties and the interactive nature of tasks, it is clear that the most appropriate way to evaluate these kind of tasks is with a human in the loop, i.e. a Visual Turing Test. Unfortunately, large-scale human-in-the-loop evaluation is limited by financial and infrastructural challenges that must be overcome by each interested research group independently. Consequently, human evaluations are rarely performed and experimental settings vary widely, limiting the usefulness of these studies in benchmarking algorithms.

We propose to fill this critical need in the community by providing a central, standardized open source platform for human-in-the-loop evaluation. To this end, we have developed the infrastructure to pair Amazon Mechanical Turk (AMT) users in real-time with artificial visual dialog agents.

3.2.1 Challenges

We had to tackle several challenges associated with building a framework for connecting human workers with agents

- **Instructions**: Since the worker don't know their roles before starting the study, they need detailed instruction and a list of Do's and Dont's for the task. Each challenge might have different instructions and therefore we give challenge organizers the flexibility to provide us with their own HTML templates.

- **Worker pool**: We need to ensure that we have a pool of good quality workers which have previous experience in doing certain tasks and have a history of high acceptance rate. We allow organizers to provide us with a list of whitelisted and blocked workers.

They can also provide a qualification test which the workers need to pass to do the evaluation tasks associated with the challenge

- **Uninterrupted back-and-forth communication**: Certain tasks like evaluating dialog agents need uninterrupted back-and-forth communication between agents and workers. However, this is not always possible since turkers might disconnect or close a HIT before finishing it. We do a lot of book-keeping to ensure that incompleted HITS are re-evaluated and turkers can reconnect with the same agent if the connection was interrupted only temporarily.

- **Gathering results**: We provide a flexible JSON based schema and APIs to fetch the results from the evaluation tasks once they are completed. These results are automatically updated on the leaderboard for each submission.

3.3 Remote Evaluation

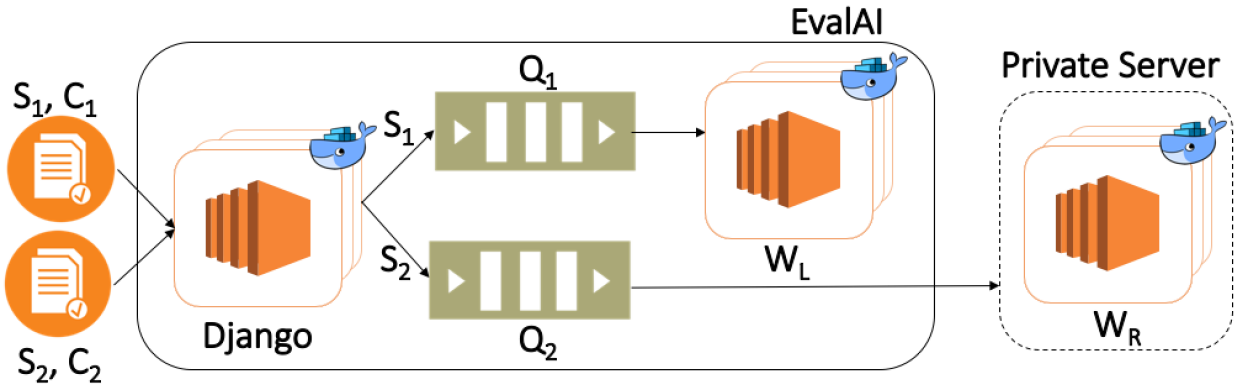


Figure 3.1: Remote Evaluation Pipeline: Challenge C_1 and C_2 are hosted on EvalAI but evaluation for C_2 happens on an evauation worker that is running on a private server which is outside EvalAI VPC. For two submissions S_1 and S_2 made to challenges C_1 and C_2 respectively, submission S_1 will be evaluated on W_1 which is running on EvalAI whereas S_2 will run on W_2 which is a remote machine.

Certain large-scale challenge need special compute capabilities for evaluation. For instance, one can imagine that running an agent based on some deep reinforcement learning

model in a dynamic environment will require powerful clusters with GPUs. While our evaluation platform might not have such powerful clusters available, it shouldn't hinder them from organizing a challenge. To solve this problem, we allow challenge organizers to spin up their own cluster of worker nodes for evaluation while we take care of hosting the challenge, handling user submissions and the maintaining the leaderboard. Our system decouples the worker nodes from the web servers through the use of message-queues. As described in Fig. 3.1, this allows us to broadcast submissions to challenge specific queues. The external pool of worker nodes will listen for incoming submission to this dedicated queue. The message payload contains all the information necessary to run evaluation on the submission and submit it to the leaderboard.

3.4 Command Line Interface

EvalAI also provides a command line interface (CLI) using which users can create, participate, and make submissions to a challenge. They can also keep track of the state-of-the-art for different challenges on EvalAI. For example, Figure 3.2 shows the leaderboard of VQA 2018 challenge on the terminal.

```
(env) ➜  ~ evalai challenge 2 leaderboard 124
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
| Rank |                   Participant Team                  | yes/no | number | other | overall |                 Last Submitted                |
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
|  1   |                        AIOZ                         | 87.96  | 54.99  | 63.28 |  72.61  |              11/14/18 11:01:03 AM             |
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
|  2   |                   HDU-UCAS-USYD                     | 87.97  | 52.51  | 63.58 |  72.49  |              07/13/18 04:52:58 PM             |
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
|  3   |                     casia_iva                       | 86.98  | 51.05  | 62.31 |  71.31  |              05/20/18 05:48:34 PM             |
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
|  4   |                   Tohoku CV Lab                     | 87.29  | 53.25  | 61.13 |  71.12  |              05/11/18 11:05:12 PM             |
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
|  5   |                      MIL-UT                         |   87   | 51.65  | 61.62 |  71.06  |              05/20/18 07:56:43 PM             |
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
|  6   |               graph-attention-network               | 86.54  | 51.65  | 61.42 |  70.77  |              05/20/18 07:45:08 PM             |
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
|  7   |                     bytedance                       | 86.83  | 49.65  | 60.96 |  70.45  |              05/20/18 10:14:22 AM             |
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
|  8   |                   TsinghuaCVLab                     |  86.7  | 50.69  | 60.75 |  70.41  |              05/31/18 12:54:47 AM             |
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
|  9   |                      DCD_ZJU                        | 86.21  | 48.82  | 61.58 |  70.4   |              05/11/18 03:50:07 AM             |
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
|  10  |                  Adelaide-Teney                     | 86.53  | 51.48  | 60.57 |  70.34  |              05/20/18 07:22:25 PM             |
+------+-----------------------------------------------------+--------+--------+-------+---------+-----------------------------------------------+
(env) ➜  ~
```

Figure 3.2: Leaderboard of Visual Question Answering 2018 Challenge on the terminal displaying top-10 teams and their accuracies for different metrics

CHAPTER 4

SYSTEM ARCHITECTURE

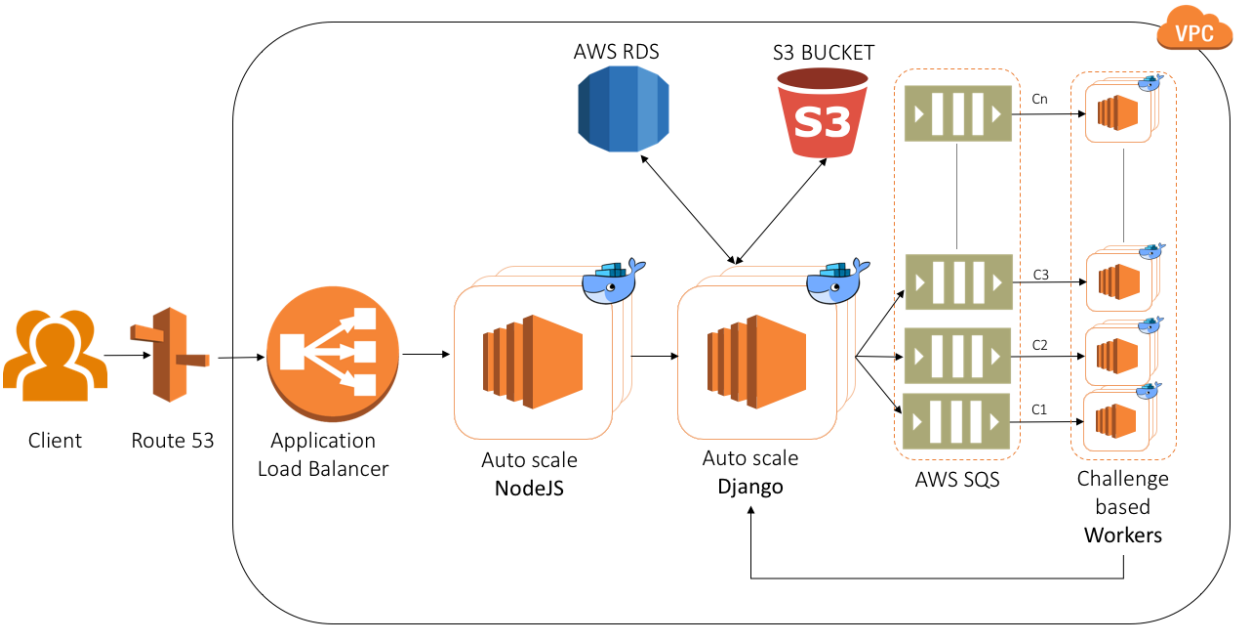


Figure 4.1: System architecture of EvalAI was designed with scalability and portability in mind from its foundations. There are four core pieces of our infrastructure: (1) the orchestration framework based on Docker and Elastic Container Service is responsible for scaling, deploying and monitoring our infrastructure to meet the requirements of all the challenges running on our system; (2) Django and Node.js based web-servers and databases which provide exhaustive APIs for challenge organizers and participants to interact with the service; (3) message queues running on Amazon Simple Queue Service (SQS) for scheduling evaluation either internally or remotely; and (4) worker nodes built on top of Docker responsible for evaluating submissions to a given challenge, evaluating agents in a dynamic environment, or setting up human-agent interactions for human evaluation.

The back-end of our system Fig. 4.1 was designed with scalability and portability in mind from its foundations. Most of the components rely heavily on open-source technologies – Docker, Django, Node.js, and Postgre-SQL – such that they can run on any cloud service provider with minimal changes. For certain components which use proprietary services of a particular cloud provider, we have tried to keep the protocol consistent with open-source alternatives. The decision to use proprietary services over open source alternatives was to

reduce the burden of maintaining those pieces of the infrastructure reliably. As a matter of fact, our open source code has been forked many times by educational institutions and industrial organizations to run their own private challenge. In the following sub-sections, we describe in detail the four key pieces of our platform: (1) **the orchestration framework** is responsible for scaling, deploying and monitoring our infrastructure to meet the requirements of all the challenges running on our system; (2) **web-servers and databases** which provide exhaustive APIs for challenge organizers and participants to interact with the service; (3) **message queues** for scheduling evaluation either internally or remotely; and (4) **worker nodes** responsible for evaluating submissions to a given challenge, evaluating agents in a dynamic environment, or setting up human-agent interactions for human evaluation.

4.1 Orchestration

Scaling up the infrastructure to serve thousands of requests is a crucial requirement for an evaluation platform that connects thousands of human workers with tens or hundreds of agents in real-time. We rely heavily on Docker [24] containers to run our infrastructure. Docker encourages a "share nothing" philosophy in which each part of the web-stack is broken down into multiple components such that each component can be scaled independently without affecting other pieces of the infrastructure. We also deploy all our containers on Amazon Elastic Container Service (ECS) [25]. which auto-scales the cluster to meet the current demands. Using container orchestration frameworks such as ECS leads to high operational efficiency. No time is spent installing softwares and maintaining a cluster manually. It greatly speeds up the time it takes to deploy new changes to the infrastructure and greatly improved our agility in deploying rapid changes to our entire infrastructure.

4.2 Web Servers

EvalAI uses Django [26] which is a high-level Python HTTP Web framework that provides a powerful interface for defining data models and REST APIs used to interact with the

services. Django is responsible for accessing and modifying the database using APIs, and submitting the evaluation requests into a queue. It additionally exposes certain APIs to serve data and fetch results from Amazon Mechanical Turk [5] during human evaluation.

EvalAI uses Node.js [27] for real-time communication between agents and workers on Amazon Mechanical Turk [5]. Node.js is an event-driven web-framework that excels in real-time applications such as chat applications. The defacto standard for building real-time Node.js applications is via Socket.IO [28]. It is an event-based bi-directional communication layer which abstracts many low-level details and transports, including AJAX long-polling and WebSockets, into a single cross-browser compatible API. Agents and workers on AMT communicate with each other using JSON blobs. By keeping the communication protocol JSON based, the challenge organizers can customize it to support any kind of interaction. Our system makes minimal assumptions about the contents of the payload and only expects metadata associated with the HIT for book-keeping purposes.

4.3 Message Queue

The message queue is responsible for routing user's submission to the appropriate worker pool based on the unique routing key associated with each challenge. For our message broker, we chose Amazon Simple Queue Service (SQS) [29]. By using SQS, we don't have to worry about consistency and reliability of the queue. The queue is robust to messages getting dropped from the queue without explicit acknowledgement by the workers. It also provides certain guarantees such as 'First-In, First-Out' and 'Exactly Once' delivery of messages. An added bonus of using SQS is that it works seamlessly with other AWS services we use.

4.4 Evaluation Worker Nodes

For every challenge, there is a different pool of worker nodes dedicated to evaluating submissions for that challenge. Designing a worker node that supports custom evaluation is hard. A key requirement for such a worker node is that it needs to be isolated from other worker

nodes that belong to different challenges. This way, the dependencies for one challenge don't clash with dependencies of other challenges. Secondly, they need to scale independently based on the demands of the challenge. Some challenge might have computationally intensive evaluations which will require that the workers scale vertically to prevent out of memory errors. While other challenges might also have too many concurrent submissions for which they need to scale horizontally. To address both these issues simultaneously, we spawn worker nodes as docker containers running inside Elastic Container Service (ECS) [25]. This keeps worker nodes isolated from each other and can be scaled independently on demand. We also provide a base image to the organizers which contains all the necessary abstraction to listen to the SQS queue for new submission, evaluate the submission and update the leaderboard database with the results. Given this abstraction, the challenge organizers simply have to write an evaluation script and are free to modify the worker to add additional dependencies they might have.

We also worked closely with challenge organizers to optimize their code to leverage the full computational capacity of the worker. For instance, unlike CodaLab which invokes a python process to evaluate a new submission every single time, we **warm-up the worker nodes at start-up** by importing the challenge code and pre-loading the dataset in memory. We also split the dataset into small chunks that are simultaneously evaluated on multiple cores. These simple tricks result in faster evaluation and reduces the evaluation time by an order of magnitude in some cases.

CHAPTER 5
LIFECYCLE OF A CHALLENGE

We now walk you through the life-cycle of a competition which involves creating a competition, submitting entries to the competition and evaluating the submissions. This will also help understand how different components of the platform communicate with each other.

5.1 Challenge Creation

Challenge creation on EvalAI is made easy by providing two ways of creating a challenge. For complex use-cases which require custom evaluation metrics, multiple dataset splits and phases, users are recommended to create a competition bundle which contains all the source code, dataset and configuration files required to host the challenge. On the other hand, if the challenge involves standard evaluation tasks like image recognition, object detection or if it uses a standard automatic metric, then the users can create the competition by following a bunch of prompts on the website.

5.1.1 Challenge configuration

The configuration file defines a bunch of useful parameters such title of the challenge, start date, end date, number of challenge phases, number of submissions allowed per day for each of these phases along with several others. It also maps each challenge phase with one of the defined dataset splits and a leader-board. Additionally, we allow users to add challenge details in the form of HTML templates or Markdown which will be displayed on the webpage when the competition goes live. These can be used to introduce the task, explain the evaluation protocol and describe the dataset to the participants. Here is a sample configuration file that list down some important fields:

```yaml
title: Sample Challenge
short_description: Description
description: desc.html
evaluation_details: eval.html
terms_and_conditions: tnc.html
image: logo.jpg
submission_guidelines: guidelines.html
evaluation_script: eval_script.zip
start_date: 2018-09-28 00:00:00
end_date: 2019-01-31 23:59:59
published: True
leaderboard:
  - id: 1
    schema: {
                "labels": ["Metric1", "Metric2"],
                "default_order_by": "Metric2"
            }
challenge_phases:
  - id: 1
    name: Test Phase
    description: test.html
    leaderboard_public: True
    is_public: True
    start_date: 2018-09-28 00:00:00
    end_date: 2019-01-31 23:59:59
    test_annotation_file: test_ann.json
    codename: test
    max_submissions_per_day: 5
    max_submissions: 50
dataset_splits:
  - id: 1
    name: Test Split
    codename: test_split
challenge_phase_splits:
  - challenge_phase_id: 1
    leaderboard_id: 1
    dataset_split_id: 1
    visibility: 1
```

Evaluation scripts associated with the challenge constitutes the next important piece of the bundle. For each challenge, organizers can package their evaluation algorithms as Docker images. Organizers can build the image on top of a base image that we provide which

contains all the necessary pieces required to connect the evaluation script with incoming submissions in production. We describe the inner workings of our evaluation workers later in the Evaluation section.

5.2 Submission

Depending on the challenge, participants can submit two types of files: (1) a file containing all the predictions on a static test set provided by the challenge organizers or (2) a docker image containing the participant's model which will be later used to evaluate on newer unseen test scenarios in the environment. Making a submission to a competition involves uploading a file in the prescribed format to the evaluation server. The user can optionally provide a small description of the method used for this submission.

5.2.1 Submitting the predictions

If the type of submission is predictions from a model, we use REST APIs along with Queue based architecture to process submissions. When a participant makes a submission for a challenge, a REST API is called which first stores the submission in a database and publishes a message to the queue with the submission metadata. The message is routed to a specific queue using the routing key corresponding to the competition. This way, a submission can be easily directed to external compute workers for remote evaluation too (see Fig. 5.1). On the other end of the queue, a worker is listening for incoming submissions to evaluate them. On successful evaluation, the leader-board will reflect the latest ratings. Additionally the the 'stderr' or 'stdout' streams from the container are also made available to the users to debug their predictions led to an error while evaluating.

5.2.2 Submitting an agent

Users can also submit their agents (Fig. 5.1) for dynamic evaluation either in simulation environments or by pairing with a real human through AMT. Challenge organizers release a

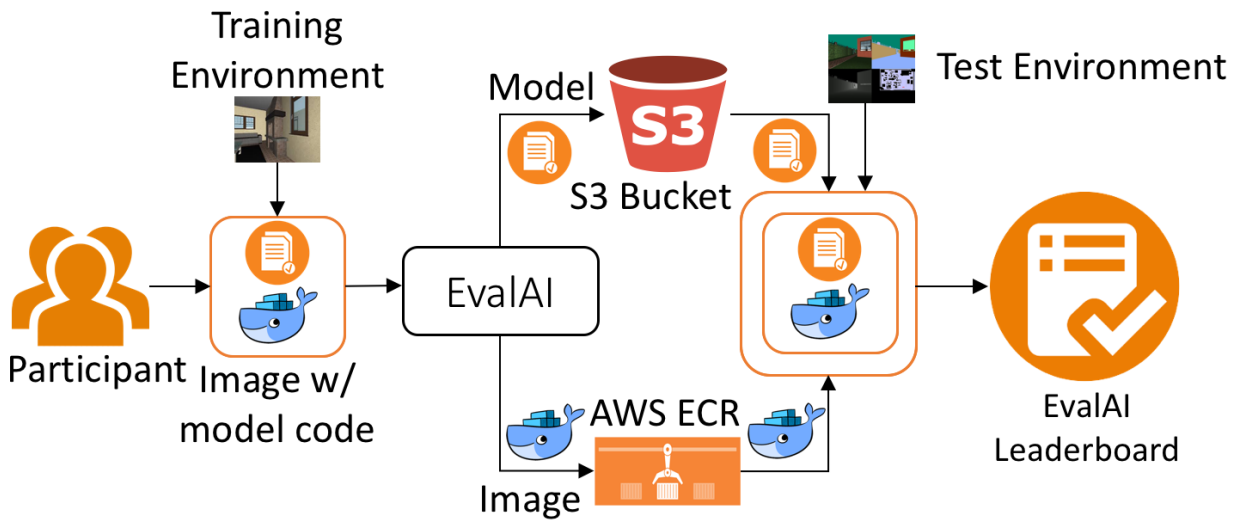


Figure 5.1: EvalAI allows participants to submit the code for their agent such that they can be evaluated in dynamic environments on the evaluation server. The pipeline involves participants submitting the code as docker image and the model snapshot to the evaluation server. These get stored in Amazon Elastic Container Registry and Amazon S3 respectively. During evaluation, the worker fetches the image, test environment and the model snapshot and spins up a new container to perform evaluation on this model. The results are then sent over to the leader-board.

base docker image which is responsible for communicating with EvalAI and Mechanical Turk. The participant is free to extend the docker image by installing additional dependencies. At the time of submission, the participant packages the code inside a docker image and upload their model and docker image on Amazon S3 and Container Registry respectively. By decoupling the model snapshot and the image containing the code to run the agent, subsequent submissions will be faster since participants won't have to upload docker images to the registry again and again. On submission, when the worker node is ready to evaluate the participant's submission, it launches a docker container based off the submitted image and attaches the model and the test environment as volume to the container. The worker will run the 'eval' script inside the container to start evaluating the task. The final scores and metrics are communicated by the worker to the central leaderboard. An added advantage of running this inside an isolated docker container is that it helps us prevent security issues. We take additional precautionary steps like preventing access to outside internet so that the

agents don't broadcast the private test set to the internet.

5.3 Evaluation

5.3.1 Automatic evaluation

Standard tasks like image recognition, object detection have well established evaluation protocol that relies on standard metrics like accuracy, precision and recall. We allow organizers to provide an implementation of their metric and is subsequently used to evaluate all submissions ensuring consistency in evaluation protocols. The by-product of containerizing the evaluation for different challenges in docker containers is that it allows us to package fairly complex pipelines, with all it's dependencies in an isolated environment. Challenge organizers have the full freedom to choose any programming language to write their evaluation script. The only requirement is a simple wrapper over their functions to make it compatible with EvalAI I/O protocol. Here is what a sample python script to evaluate may look like.

```
def evaluate(test_annotation_file, user_annotation_file, phase_codename, **kwargs):
  submission_metadata = kwargs.get("submission_metadata")
  # Do stuff here

  score = process_submission(user_annotation_file,
    test_annotation_file)

  if score > 90:
    slack_data = kwargs.get("submission_metadata")
    webhook_url = "Slack Webhook URL"

    response = requests.post(
        webhook_url,
        data=json.dumps({'text': "*Flag raised for submission:* \n" + str(slack_data)}),
        headers={'Content-Type': 'application/json'},)
```

Once the competition is live, our system mounts the dataset directory to the evaluation container and run a worker that is listening to the queue to fetch the incoming submissions. The payload fetched from the message queue contains all useful metadata such as path to

input submission file, challenge phase details that might be useful to the evaluation script.

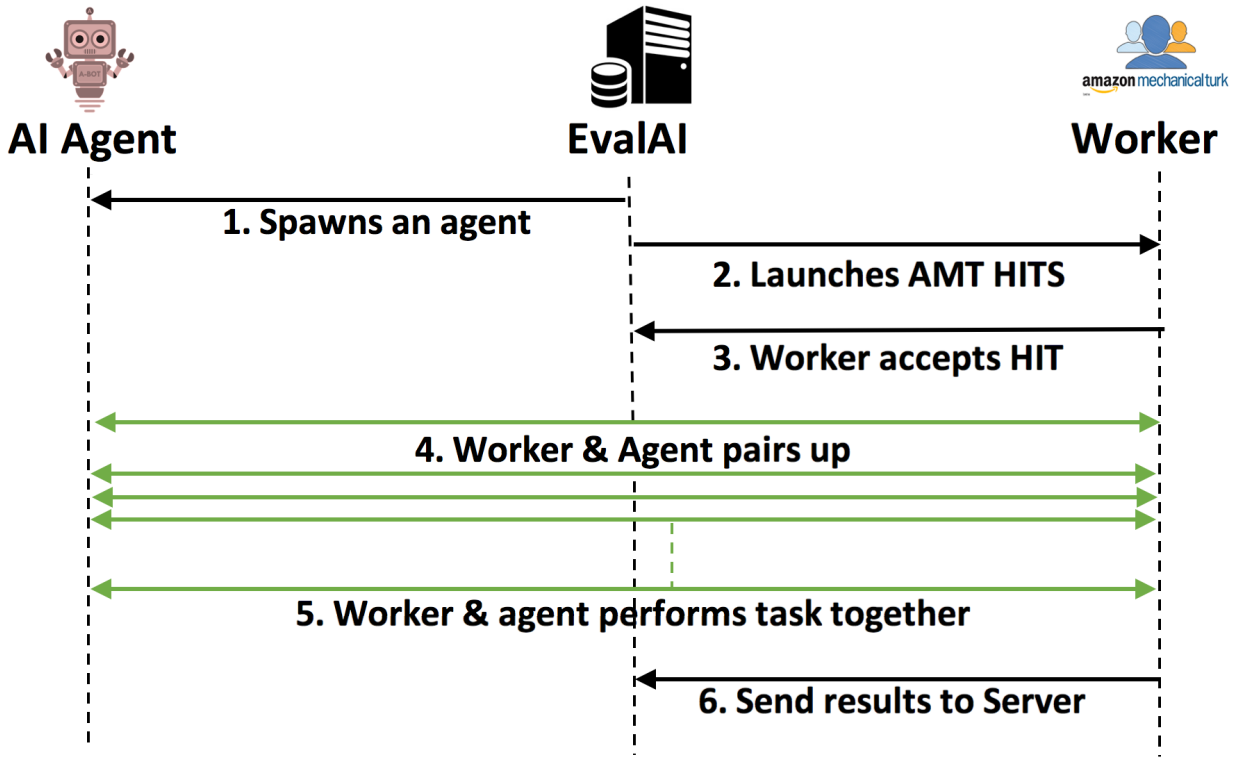


Figure 5.2: The connection protocol between an AI agent and a human worker on Amazon Mechanical Turk (AMT). On receiving the submission, EvalAI first spawns an agent and launches a corresponding HIT on AMT. Once the HIT gets accepted, EvalAI pairs up the agent with the worker. The human-worker team perform the task together for n-rounds. At the end of this interaction, the worker rates the AI agent which is then sent to the leader-board.

5.3.2 Human-in-the-loop evaluation

For human-in-the-loop evaluation (5.2), the evaluation code first loads up the worker and launches a new HIT on Amazon Mechanical Turk. Once the worker accepts the HIT, the worker is paired with the agent running inside a docker image. Based on the instruction given, the worker will interact with the agent and evaluate it according to certain criteria. This interaction data and the final rating given by the worker is stored by EvalAI which is eventually reflected on the leaderboard. EvalAI takes care of managing a persistent con-

nection between the agent and the worker, error handling, retrying , storing the interaction data corresponding to this HIT and automatically approving or rejecting HIT. By building a generic interface to model human-robot interactions, we make it extremely simple for large-scale human evaluation of tasks that cannot be appropriately evaluated using automatic metrics. We discuss one human-in-the-loop task in the second case study.

5.4 Webhook and Notification

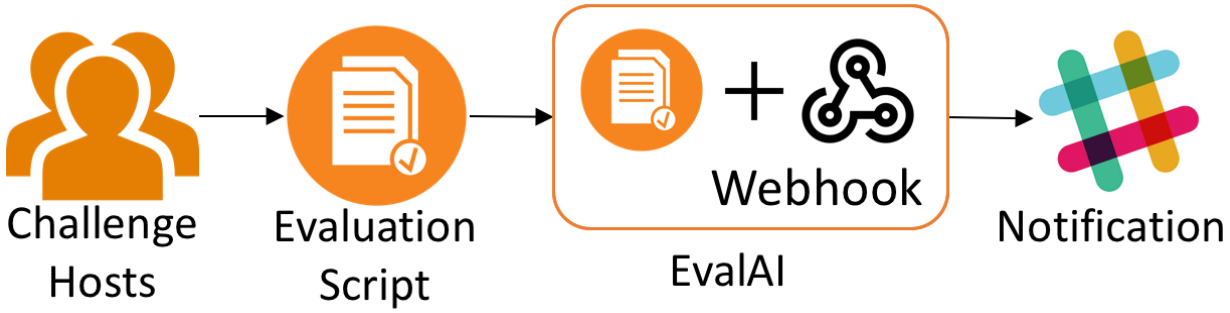


Figure 5.3: Pipeline demonstrating integration with third party webhook services to track progress of a challenge submission at hosts end.

We also provide webhooks that provide a simple way to post messages from EvalAI to different apps. The challenge organizers can be notified of certain activities likes new submissions while participants can be notified when a new challenge has been launched.

CHAPTER 6
CASE STUDIES

In this section, we go over several past challenges and proof-of-concept challenges organized on our platform to showcase its various capabilities. We describe three concrete instances where EvalAI offers substantial improvements over other existing systems, specifically demonstrating significant speedup over existing platforms while supporting human evaluation for simple multimodal as well as completely free-form agents.

6.1 Visual Question Answering

Visual Question Answering (VQA) is a multi-modal task where given an image and a free-form open-ended natural language question about the image, the AI agent's task is to answer the question accurately. The VQA Challenge 2016 was organized on the first version of the associated dataset and was hosted on another platform, where on average each individual evaluation took approximately 10 minutes to complete. However, the team switched to EvalAI to host the VQA Challenge in 2017 and 2018 (6.2) which was organized on the second version of the dataset – twice as large as v1. Even though the dataset for the VQA Challenge 2017 doubled, we found that our parallel implementation offered a significant reduction in evaluation time, taking only ~130 seconds per submission to evaluate on the whole test split. This was made possible by leveraging map-reduce techniques to distribute smaller chunks of the dataset on multiple cores and eventually combining the results to calculate the accuracy of the entire dataset. Additionally, we don't require that program gets loaded everytime a new submission is made for the competition. By keeping a list of workers ready to evaluate as soon as a new submission comes in, the latency caused due to reloading the evaluation code again and again is prevented.

The challenge also used several other features of our platform, such as creation of mul-

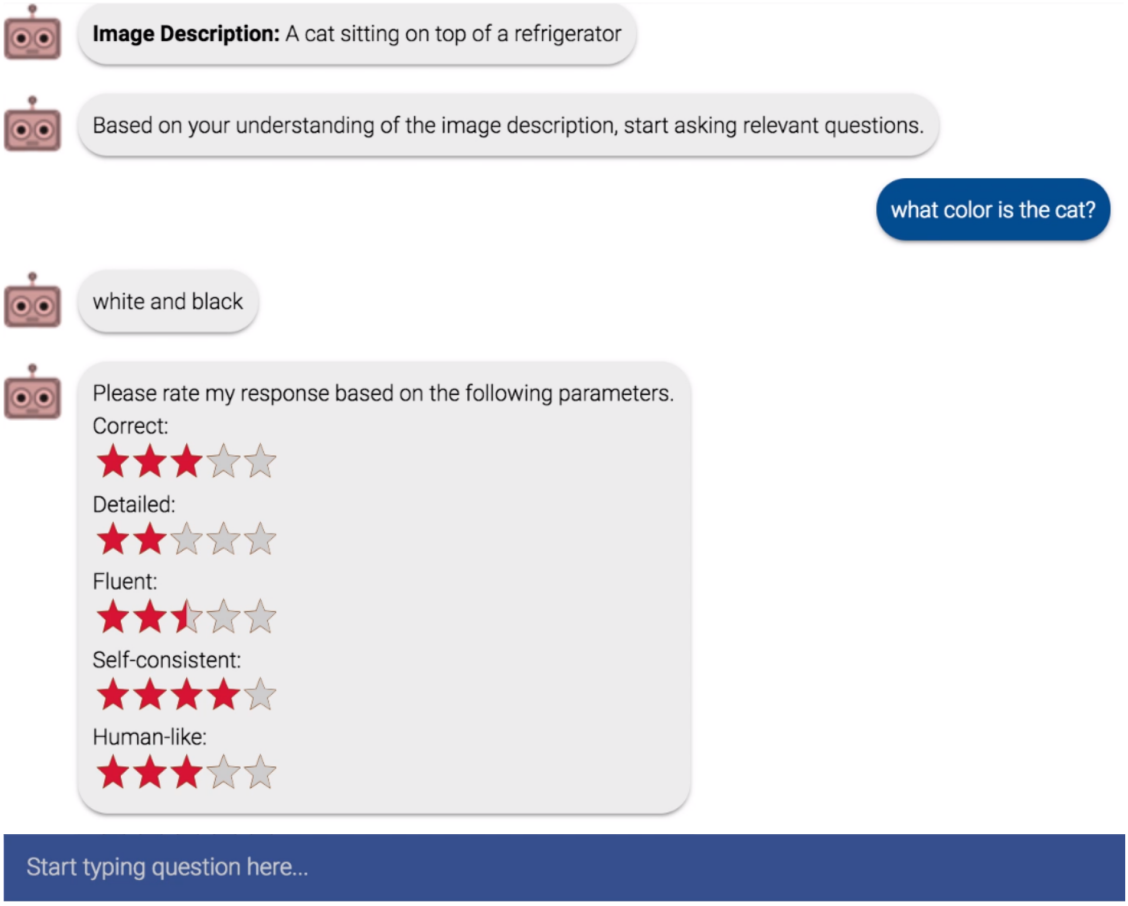


Figure 6.1: Human-in-the-loop interface for evaluating visual dialog agents. In this task, humans are paired with agents that have free-form dialog about an image in the form of sequential question answering. Humans are tasked with providing correct answers to the questions asked by agent. The agent remembers the conversation and ask follow-up questions. Based on this conversation, humans have to evaluate the agent on several metrics like correctness, fluency, consistency etc.

tiple challenge phases and having multiple data-splits. `test-dev` is used for debugging and validation experiments and allows for unlimited submission to the evaluation server; `test-standard` is the 'default test' data for the competition When comparing to the state of the art (e.g., in papers), results are reported on `test-standard`. Test-standard is also used to maintain a public leaderboard that is updated upon submission. One might also have a Test-reserve split used to protect against possible over-fitting. If there are substantial differences between a method's scores on `test-standard` and `test-reserve`, this raises a red-flag and prompts further investigation. Results on test-reserve are not publicly revealed.

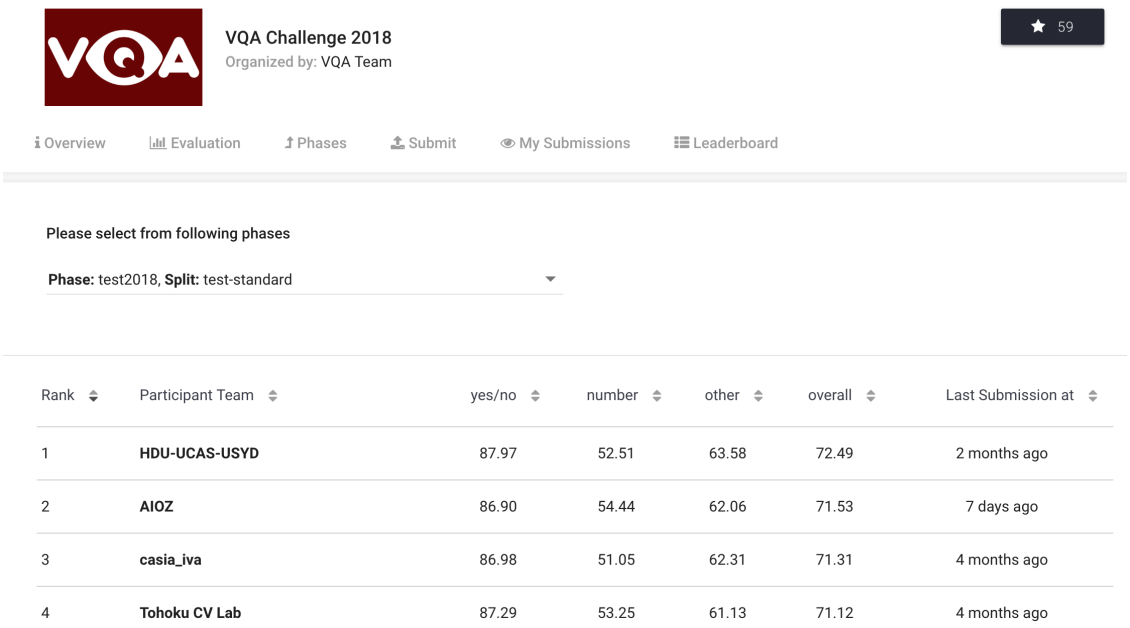


Figure 6.2: : Screenshot of the VQA Challenge 2018 Leaderboard. EvalAI allows challenge organizers to report a variety of number of metrics like yes/no, number, other, overall VQA accuracy. There were over 7000 submissions made by approximately 150 teams in the 2018 version of the challenge.

Finally, `test-challenge` is used to determine the winners of the challenge. The leaderboards also have different visibility. Public leaderboards (corresponding to test-dev and test-std) are publicly visible to everyone. The leaderboard corresponding to test-challenge, on the other hand, is only available after the submission phase is over and the results are announced. It is however always visible to the challenge organizers.

6.2 Visual Dialog

While evaluation via automatic metrics is straightforward, for tasks where automated metrics are not truly reflective of a holistic sense of improvement – supporting human evaluation is a necessity. As mentioned before, it is notoriously difficult to evaluate natural language generation tasks such as image captioning [21, 22], visual dialog [10, 23, 30] etc using automatic metrics. Developing measures which correlate well with human judgment remains an open area of research. Focusing on Visual Dialog, where given an image, an associated

dialog history and a follow-up question about the image, an agent is required to answer the question while grounding the question in the history – evaluation is further complicated by the huge set of possibly 'correct' answers and the relatively sparse sampling of this space, even in large-scale datasets. Given these difficulties and the interactive nature of the task, it is clear that the most appropriate way to evaluate dialog agents is with a human in the loop, i.e. a visual Turing Test. As part of a demonstration at CVPR 2018, we hosted a visual dialog challenge on EvalAI (6.1) where each submission was connected with a human subject on Amazon Mechanical Turk tasked with evaluating a response generated by participating model. After 10 such rounds of human-agent interaction, the human's rating of the agent was reflected as a score on the EvalAI leaderboard in real-time.

6.3 Embodied Question Answering

Finally, as we move towards developing intelligent agents for tasks situated in *active environments* instead of *static datasets*, where agents take actions to change the state of the world around them, it is imperative that we build new tools to accurately benchmark agents in environments. One example of such a task is Embodied Question Answering [8] – an agent is spawned at a random location in a simulated environment (say in a kitchen) and is asked a natural language question ("What color is the car?"). The agent perceives its environment through first-person vision and can perform a few actions: {`move-forward, turn-left, turn-right, stop`}. The agent's objective is to explore the environment and gather visual information necessary to answer the question ("orange"). Evaluating agents for EmbodiedQA presents a key challenge – instead of a hidden test dataset, there are hidden test environments, so participants have to submit pretrained models and inference code, which has to be reliably executed in these environments to benchmark them. In ongoing work, we have developed an evaluation framework for EmbodiedQA – wherein participants upload Docker images with their pretrained models on Amazon S3, which is then attached and run against test environments and evaluation metrics provided by the challenge organizer.

CHAPTER 7

IMPACT ON COMMUNITY

From the very start, EvalAI was built for the students, researchers, data-scientists with the aim to create, collaborate, participate in Artificial Challenges organized around the globe. In this regard, we have open sourced the whole codebase of EvalAI[1] on Github under the CloudCV [31] Organization[2]. This allowed to build an open source community around EvalAI. We discuss more about the open source community in the following section.

7.1 Open Source Community

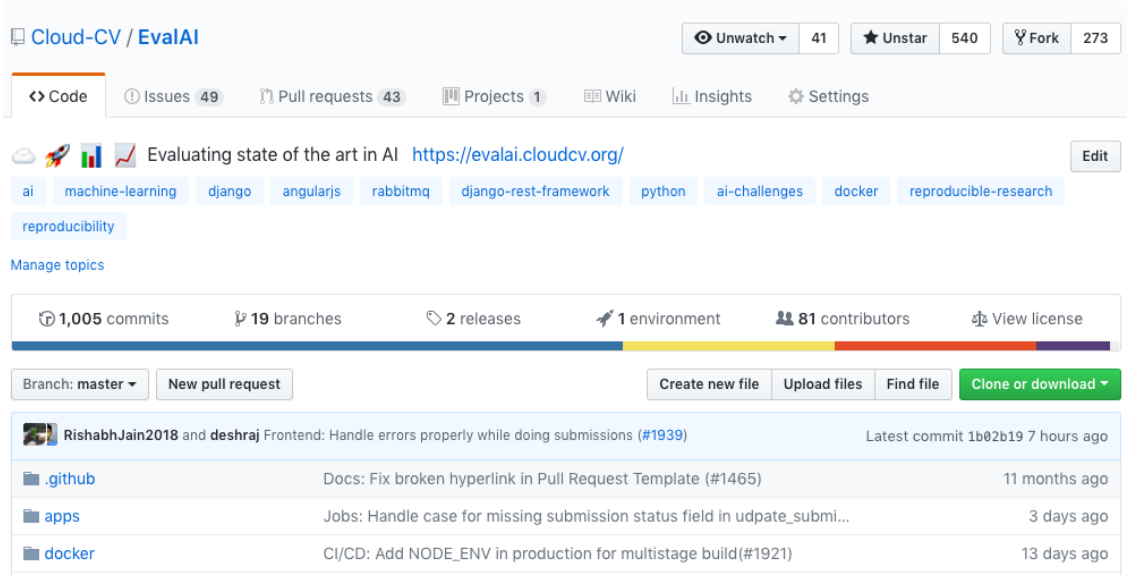


Figure 7.1: GitHub Page of EvalAI Project

After open sourcing the code base on GitHub, we received a lot of attention from the open source community on GitHub. Several developers and designers were interested in contributing to EvalAI. We received a lot of love from open source community and now we

[1] `https://github.com/Cloud-CV/EvalAI`
[2] `https://github.com/Cloud-CV`

have 80+ contributors, 1900+ Pull Requests and Issues, 500+ Stars on repository, and 270+ forks (user creating a copy of the repository to add their own changes to the repository).

7.2 Google Summer of Code (GSoC) and Google Code-In (GCI)

Google Summer of Code (GSoC) is a global program focused on bringing more student developers into open source software development. Students work on a three month programming project with an open source organization during their break from university. GSoC attracts thousand of smart and enthusiastic students from around the world that spend their summer vacations working on an open source project under the guidance of mentors from participating organizations.

Similarly, Google Code-In is organized by Google where pre-university students ages 13 to 17 are invited to take part in the global, online contest where they are introduced to a wide variety of bite-sized tasks, which are easy for beginners to jump in and get started.

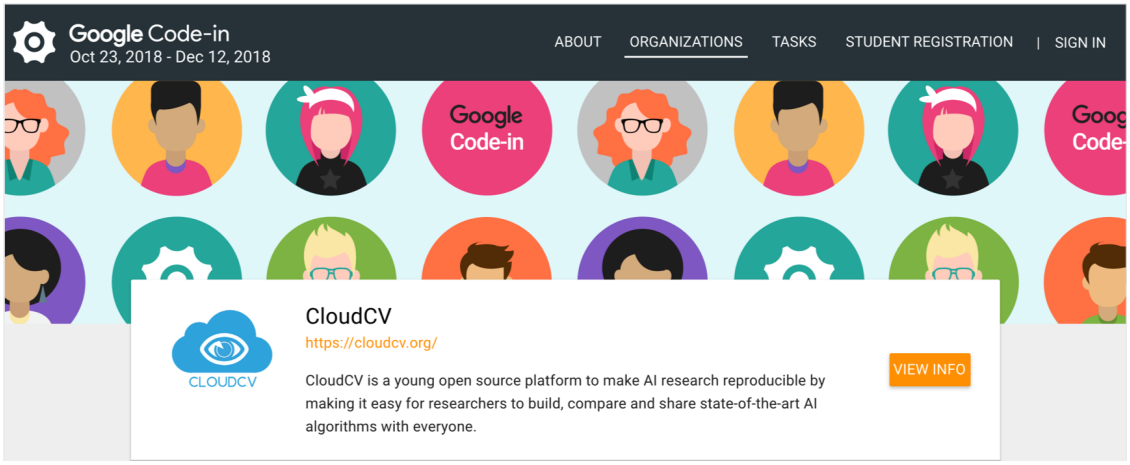


Figure 7.2: Google Code-In 2018 page listing CloudCV as one of the selected open source organization

EvalAI was selected in Google Summer of Code and Google Code-In as an open source project under the CloudCV [31] Organization for two consecutive years (2017 and 2018). We had this amazing opportunity to mentor 3 university students and several pre-university students during GSoC and GCI.

7.3 AI Community

EvalAI has been hosting challenges from different domains such as Robotics, Machine Learning, Natural Language Processing, and Computer Vision etc. Currently, we have more than 2,100 registered users, hosted 12+ challenges with participation from more than 900 teams. EvalAI has received more than 15,000 submissions in total. Currently, we have more than 1,200 monthly active users on our platform which is growing rapidly. 7.3 shows the demographics of the users on EvalAI from 120 different countries.

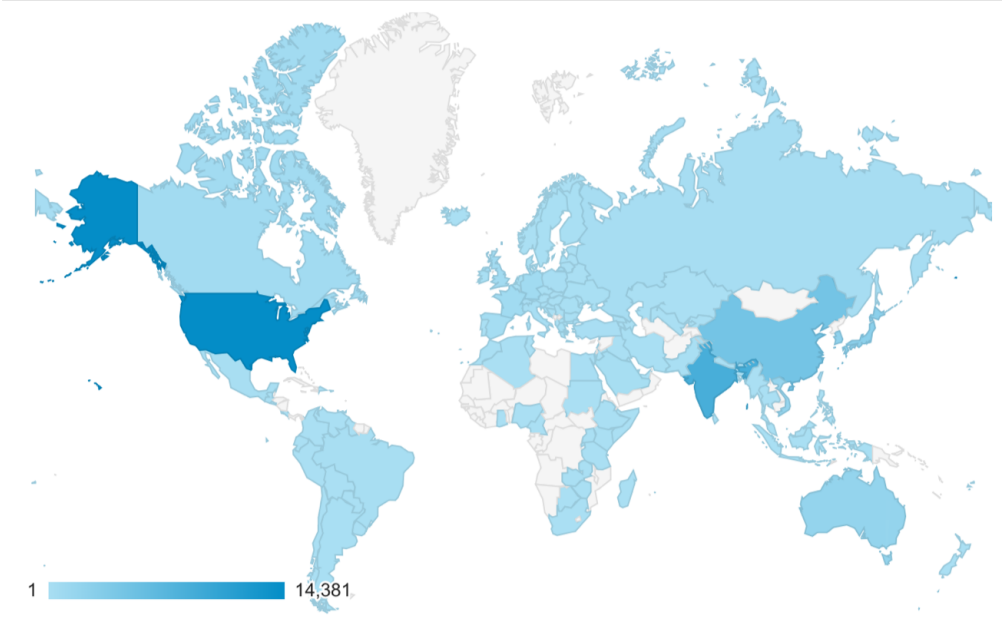


Figure 7.3: Heat map visualization showing registered users from 120 different countries around the world

CHAPTER 8
CONCLUSION AND FUTURE WORK

8.1 Conclusion

In this thesis, we presented EvalAI, an open source platform for students, researchers and data-scientists to create, collaborate and participate in the Artificial Intelligence Challenges that are organized around the globe. Exhaustive and proper benchmarking of models have led to tremendous progress in the field of AI in the past and will continue to do so in the forseeable future. The overarching goal of building EvalAI is to provide the right tools, infrastructure and framework to setup exhaustive evaluation protocols for both traditional static evaluation tasks as well as those in dynamic environments hosting multiple agents and/or humans.

EvalAI offers features like human-in-the-loop evaluation, evaluation inside environments, support for custom metrics and phases, remote evaluation etc that enables challenge organizers to customize the challenge according to their requirement. EvalAI runs on a group of machines running on Amazon Web services capable of running evaluating submissions to different challenges at the same time in a distributed fashion using services like AWS Elastic Container Service (ECS). Participants can make submissions to challenges through the website, REST APIs or using command line interface that we provide. It also enables participants to upload docker images through the interactive website user interface. For larger challenges such as Visual Question Answering Challenge, the system is capable to scale up and down automatically based on the load on a particular challenge. EvalAI also acts as a homework assignment platform where professors can host deep learning and computer vision challenges for their classrooms that helps students to build and compare their models against the existing models out there. To further extend the reach of EvalAI, we open sourced the entire

code base of EvalAI which allowed us to build a symbiotic relationship with the open source community. We also participated in one of the most coveted and highly selective open source programs called the Google Summer of Code (GSOC) and Google Code-In in 2017 and 2018. Through these programs, we are closely working with the students to further develop the features of EvalAI in order to make the challenge creation and participation process much easier and help in lower the barrier to entry in the field of Artificial Intelligence.

8.2 Future Work

8.2.1 Human-in-the-loop Evaluation for different tasks

While traditional evaluation platforms were adequate for evaluation of tasks like image classification, scene recognition using automatic metrics, there is a critical need to support human evaluation of tasks like dialog, question answering, and captioning etc. To this end, we have developed a new evaluation platform that allows large-scale human-in-the-loop evaluation for Visual Dialog Agents. The system can pair up an agent with thousand of workers that can evaluate the agent based on their interaction. This will eventually help us understand the agents better and improve performance of agents both in isolation and in human-AI teams.

We want to extend this support of human-in-the-loop evaluation for other tasks as well. In the end, we aim to release our own Javascript library that contains templates for different kinds of tasks to do human-in-the-loop evaluation. Challenge organizers will have the capability to modify the templates according to their requirements. We also want to make this feature more robust since there are certain corner cases that we will figure out as we test the feature with other AI tasks.

8.2.2 Interactive demos

Studies in [32, 33] have shown that looking at model's prediction in an interactive demo enable researchers to identify failure modes. Therefore, we want to enable researchers to

spin up interactive demos from their model's prediction without prior web development experience.

By relying on technologies like Docker, we enable evaluation of agents in dynamic environment as opposed to static datasets. We plan to build feature that enables participants to create a demo out of their agent code without having prior knowledge of web technologies such as HTML, CSS, Javascript and some backend web framework such as Flask, Django etc required to create an online demo. This will help the community see when a particular agent fails and will help to develop better model architectures and hence will help to push state-of-the-art for different AI tasks.

Appendices

APPENDIX A
EVALUATING VISUAL CONVERSATIONAL AGENTS VIA COOPERATIVE HUMAN-AI GAMES

As AI continues to advance, human-AI teams are inevitable. However, progress in AI is routinely measured in isolation, without a human in the loop. It is crucial to benchmark progress in AI, not just in isolation, but also in terms of how it translates to helping humans perform certain tasks, *i.e.*, the performance of human-AI teams.

In this work, we design a cooperative game – GuessWhich – to measure human-AI team performance in the specific context of the AI being a visual conversational agent. GuessWhich involves live interaction between the human and the AI. The AI, which we call ALICE, is provided an image which is unseen by the human. Following a brief description of the image, the human questions ALICE about this secret image to identify it from a fixed pool of images.

We measure performance of the human-ALICE team by the number of guesses it takes the human to correctly identify the secret image after a fixed number of dialog rounds with ALICE. We compare performance of the human-ALICE teams for two versions of ALICE. Our human studies suggest a counter-intuitive trend – that while AI literature shows that one version outperforms the other when paired with an AI questioner bot, we find that this improvement in AI-AI performance does not translate to improved human-AI performance. This suggests a mismatch between benchmarking of AI in isolation and in the context of human-AI teams.

A.1 Introduction

As Artificial Intelligence (AI) systems become increasingly accurate and interactive (*e.g.* Alexa, Siri, Cortana, Google Assistant), human-AI teams are inevitably going to become more commonplace. To be an effective teammate, an AI must overcome the challenges in-

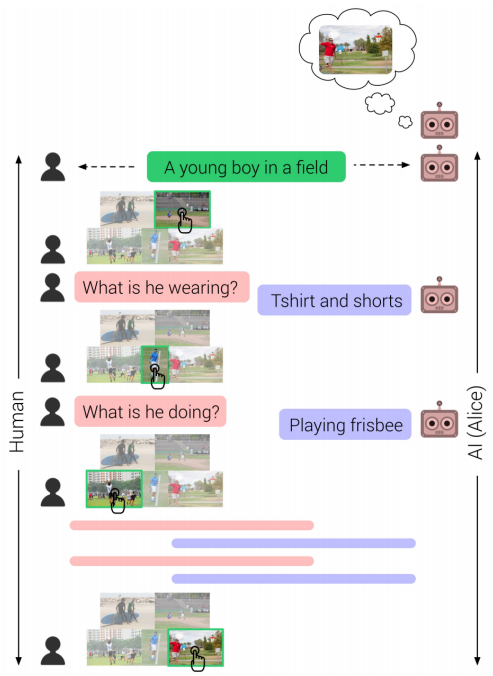


Figure A.1: A human and an AI (a visual conversation agent called ALICE) play the proposed GuessWhich game. At the start of the game (top), ALICE is provided an image (shown above ALICE) which is unknown to the human. Both ALICE and the human are then provided a brief description of the image. The human then attempts to identify the secret image. In each subsequent round of dialog, the human asks a question about the unknown image, receives an answer from ALICE, and makes a best guess of the secret image from a fixed pool of images. After 9 rounds of dialog, the human makes consecutive guesses until the secret image is identified. The fewer guesses the human needs to identify the secret image, the better the human-AI team performance.

volved with adapting to humans; however, progress in AI is routinely measured in isolation, without a human in the loop. In this work, we focus specifically on the evaluation of visual conversational agents and develop a human computation game to benchmark their performance as members of human-AI teams.

Visual conversational agents [10, 23, 34, 35] are AI agents trained to understand and communicate about the contents of a scene in natural language. For example, in Fig. A.1, the visual conversational agent (shown on the right) replies to answers questions about a scene while inferring context from the dialog history – Human: "What is he doing?" Agent: "Playing frisbee". These agents are typically trained to mimic large corpora of human-human dialogs and are evaluated automatically on how well they retrieve actual human responses

(ground truth) in novel dialogs.

Recent work has evaluated these models more pragmatically by evaluating how well pairs of visual conversational agents perform on goal-based conversational tasks rather than response retrieval from fixed dialogs. Specifically, [23] train two visual conversational agents – a questioning bot QBOT, and an answering bot ABOT – for an image-guessing task. Starting from a description of the scene, QBOT and ABOT converse over multiple rounds of questions (QBOT) and answers (ABOT) in order to improve QBOT's understanding of a secret image known only to ABOT. After a fixed number of rounds, QBOT must guess the secret image from a large pool and both QBOT and ABOT are evaluated based on this guess.

[23] compare supervised baseline models with QBOT-ABOT teams trained through reinforcement learning based self-talk on this image-guessing task. They find that the AI-AI teams improve significantly at guessing the correct image after self-talk updates compared to the supervised pretraining. While these results indicate that the self-talk fine-tuned agents are better visual conversational agents, crucially, it remains unclear if these agents are indeed better at this task when *interacting with humans*.

GuessWhich. In this work, we propose to evaluate if and how this progress in AI-AI evaluation translates to the performance of human-AI teams. Inspired by the popular GuessWhat or 20-Questions game, we design a human computation game – GuessWhich – which requires collaboration between human and visual conversational AI agents. Mirroring the setting of [23], GuessWhich is an image-guessing game that consists of 2 participants – *questioner* and *answerer*. At the start of the game, the answerer is provided an image that is unknown to the questioner and both questioner and answerer are given a brief description of the image content. The questioner interacts with the answerer for a fixed number of rounds of question-answer (dialog) to identify the secret image from a fixed pool of images (see Fig. A.1).

We evaluate human-AI team performance in GuessWhich, for the setting where the questioner is a human and the answerer is an AI (that we denote ALICE). Specifically, we evaluate two versions of ALICE for GuessWhich:

1. ALICE$_{\texttt{SL}}$ which is trained in a supervised manner on the Visual Dialog dataset [10] to mimic the answers given by humans when engaged in a conversation with other humans about an image, and

2. ALICE$_{\texttt{RL}}$ which is pre-trained with supervised learning and fine-tuned via reinforcement learning for an image-guessing task as in [23].

It is important to appreciate the difficulty and sensitivity of the GuessWhich game as an evaluation tool – agents have to understand human questions and respond with accurate, consistent, fluent and informative answers for the human-AI team to do well. Furthermore, they have to be robust to their own mistakes, *i.e.*, if an agent makes an error at a particular round, that error is now part of its conversation history, and it must be able to correct itself rather than be consistently inaccurate. Similarly, human players must also learn to adapt to ALICE's sometime noisy and inaccurate responses.

At its core, GuessWhich is a game-with-a-purpose (GWAP) that leverages human computation to evaluate visual conversational agents. Traditionally, GWAP [36] have focused on *human-human collaboration*, *i.e.* collecting data by making humans play games to label images [37], music [38] and movies [39]. We extend this to human-AI teams and to the best of our knowledge, our work is the first to evaluate visual conversational agents in an interactive setting where humans are continuously engaging with agents to succeed at a cooperative game.

Contributions. More concretely, we make the following contributions in this work:

- We design an interactive image-guessing game (GuessWhich) for evaluating human-AI team performance in the specific context of the AIs being visual conversational agents. GuessWhich pairs humans with ALICE, an AI capable of answering a sequence of questions about images. ALICE is assigned a secret image and answers questions asked about that image from a human for 9 rounds to help them identify the secret image.

- We evaluate human-AI team performance on this game for both supervised learning (SL) and reinforcement learning (RL) versions of ALICE. Our main experimental finding is that despite significant differences between SL and RL agents reported in previous work [23], we find *no significant difference* in performance between $\text{ALICE}_{\texttt{SL}}$ or $\text{ALICE}_{\texttt{RL}}$ when paired with human partners. This suggests that while self-talk and RL are interesting directions to pursue for building better visual conversational agents, there appears to be a disconnect between AI-AI and human-AI evaluations – progress on former does not seem predictive of progress on latter. This is an important finding to guide future research.

A.2 Related Work

Given that our goal is to evaluate visual conversational agents through a human computation game, we draw connections to relevant work on visual conversational agents, human computation games, and dialog evaluation below.

Visual Conversational Agents. Our AI agents are visual conversational models, which have recently emerged as a popular research area in visually-grounded language modeling [10, 23, 34, 35]. [10] introduced the task of Visual Dialog and collected the VisDial dataset by pairing subjects on Amazon Mechanical Turk (AMT) to chat about an image (with assigned roles of questioner and answerer). [23] pre-trained questioner and answerer agents on this VisDial dataset via supervised learning and fine-tuned them via self-talk (reinforcement learning), observing that RL-fine-tuned QBOT-ABOT are better at image-guessing after interacting with each other. However, they do not evaluate if this change in QBOT-ABOT performance translates to human-AI teams.

Human Computation Games. Human computation games have been shown to be time- and cost-efficient, reliable, intrinsically engaging for participants [40, 41], and hence an effective method to collect data annotations. There is a long line of work on designing such Games with a Purpose (GWAP) [36] for data labeling purposes across various domains in-

cluding images [37, 42, 43, 44], audio [45, 38], language [46, 47], movies [39] *etc*. While such games have traditionally focused on human-human collaboration, we extend these ideas to human-AI teams. Rather than collecting labeled data, our game is designed to measure the effectiveness of the AI in the context of human-AI teams.

Evaluating Conversational Agents. Goal-driven (non-visual) conversational models have typically been evaluated on task-completion rate or time-to-task-completion [48], so shorter conversations are better. At the other end of the spectrum, free-form conversation models are often evaluated by metrics that rely on n-gram overlaps, such as BLEU, METEOR, ROUGE, but these have been shown to correlate poorly with human judgment [49]. Human evaluation of conversations is typically in the format where humans rate the quality of machine utterances given context, without actually taking part in the conversation, as in [23] and [50]. To the best of our knowledge, we are the first to evaluate conversational models via team performance where humans are continuously interacting with agents to succeed at a downstream task.

Turing Test. Finally, our GuessWhich game is in line with ideas in [51], re-imagining the traditional Turing Test for state-of-the-art AI systems, taking the pragmatic view that an effective AI teammate need not appear human-like, act or be mistaken for one, provided its behavior does not feel jarring or baffle teammates, leaving them wondering not about what it is thinking but whether it is.

Next, we formally define the AI agent Alice, describe the GuessWhich game setup, and present results and analysis from human studies.

A.3 The AI: Alice

Recall from Section **??** that our goal is to evaluate how progress in AI measured through automatic evaluation translates to performance of human-AI teams in the context of visual conversational agents. Specifically, we are considering the question-answering agent Abot from [23] as Abot is the agent more likely to be deployed with a human partner in real

Figure A.2: GuessWhich Interface: A user asks a question to ALICE in each round and ALICE responds with an answer. The user then selects an appropriate image which they think is the secret image after each round of conversation. At the end of the dialog, user successively clicks on their best guesses until they correctly identify the secret image.

applications (*e.g.* to answer questions about visual content to aid a visually impaired user). For completeness, we will review this work in this section.

[23] formulate a self-supervised image-guessing task between a questioner bot (QBOT) and an answerer bot (ABOT) which plays out over multiple rounds of dialog. At the start of the task, QBOT and ABOT are shown a one sentence description (*i.e.* a caption) of an image (unknown to QBOT). The pair can then engage in question and answer based dialog for a fixed number of iterations after which QBOT must try to select the secret image from a pool. The goal of the QBOT-ABOT team is two-fold, QBOT should: 1) build a mental model of the unseen image purely from the dialog and 2) be able to retrieve that image from a line-up of images.

Both QBOT and ABOT are modeled as Hierarchical Recurrent Encoder-Decoder neural networks [10, 52] which encode each round of dialog independently via a recurrent neural network (RNN) before accumulating this information through time with an additional RNN

(resulting in hierarchical encoding). This representation (and a convolutional neural network based image encoding in ABOT's case) are used as input to a decoder RNN which produces an agent's utterance (question for QBOT and answer for ABOT) based on the dialog (and image for ABOT). In addition, QBOT includes an image feature regression network that predicts a representation of the secret image based on dialog history. We refer to [23] for complete model details.

These agents are pre-trained with supervised dialog data from the VisDial dataset [10] with a Maximum Likelihood Estimation objective. This pre-training ensures that agents can generally recognize objects/scenes and utter English. Following this, the models are fine-tuned by 'smoothly' transitioning to a deep reinforcement learning framework to directly improve image-guessing performance. This annealed transition avoids abrupt divergence of the dialog in face of an incorrect question-answer pair in the QBOT-ABOT exchange. During RL based self-talk, the agents' parameters are updated by gradients corresponding to rewards depending on individual good or bad exchanges. We refer to the baseline supervised learning based ABOT as $\text{ALICE}_{\texttt{SL}}$ and the RL fine-tuned bot as $\text{ALICE}_{\texttt{RL}}$. [23] found that the AI-AI pair succeeds in retrieving the correct image more often after being fine-tuned with RL. In the following section, we outline our GuessWhich game designed to evaluate whether this improvement between $\text{ALICE}_{\texttt{SL}}$ and $\text{ALICE}_{\texttt{RL}}$ in automatic metrics translates to human-AI collaborations.

A.4 Our GuessWhich Game

We begin by describing our game setting; outlining the players and gameplay mechanics. A video of an example game being played can be found at `https://vimeo.com/229488160`.

Players. We replace QBOT in the AI-AI dialog with humans to perform a collaborative task of identifying a secret image from a pool. In the following, we will refer to ABOT as ALICE and the human player as H. We evaluate two versions of ALICE – $\text{ALICE}_{\texttt{SL}}$ and $\text{ALICE}_{\texttt{RL}}$, where SL and RL correspond to agents *trained in a supervised setting* and *fine-tuned with*

reinforcement learning respectively.

Gameplay. In our game setting, ALICE is assigned a secret image I^c (unknown to H) from a pool of images $\mathbb{I} = \{I_1, I_2, ..., I_n\}$ taken from the COCO dataset [53]. Prior to beginning the dialog, both ALICE and H are provided a brief description (*i.e.* a caption) of I^c generated by Neuraltalk2 [54], an open-source implementation of [55]. H then makes a guess about the secret image by selecting one from the pool $\mathbb{I}$ based only on the caption, *i.e.* before the dialog begins.

In each of the following rounds, H asks ALICE a question q_t about the secret image I^c in order to better identify it from the pool and ALICE responds with an answer a_t. After each round, H must select an image I^t that they feel is most likely the secret image I^c from pool $\mathbb{I}$ based on the dialog so far. At the end of $k = 9$ rounds of dialog, H is asked to successively click on their best guess. At each click, the interface gives H feedback on whether their guess is correct or not and this continues until H guesses the true secret image. In this way, H induces a partial ranking of the pool up to the secret image based on their mental model of I^c from the dialog.

A.4.1 Pool Selection

When creating a pool of images, our aim is to ensure that the game is challenging and engaging, and not too easy or too hard. Thus, we construct each pool of images $\mathbb{I}$ in two steps – first, we choose the secret image I^c, and then sample similar images as distractors for I^c. Fig. A.2 shows a screenshot of our game interface including a sample image pool and chat.

Secret Image Selection. VisDial v0.5 is constructed on $68k$ COCO images which contain complex everyday scenes with 80 object categories. ABOT is trained and validated on VisDial v0.5 *train* and *val* splits respectively. As the images for both these splits come from COCO-train, we sample secret images and pools from COCO-validation to avoid overlap.

To select representative secret images and diverse image pools, we do the following. For

each image in the COCO validation set, we extract the penultimate layer ('fc7') activations of a standard deep convolutational neural network (VGG-19 from [16]). For each of the 80 categories, we average the embedding vector of all images containing that category. We then pick those images closest to the mean embeddings, yielding 80 candidates.

Generating Distractor Images. The distractor images are designed to be semantically similar to the secret image I^c. For each candidate secret image, we created 3 concentric hyper-spheres as euclidean balls (of radii increasing in arithmetic progression) centered on the candidate secret image in fc7 embedding space, and sampled images from each sphere in a fixed proportion to generate a pool corresponding to the secret image. The radius of the largest sphere was varied and manually validated to ensure pool difficulty. The sampling proportion can be varied to generate pools of varying difficulty. Of the 80 candidate pools, we picked 10 that were of medium difficulty based on manual inspection.

A.4.2 Data Collection and Player Reward Structure

We use AMT to solicit human players for our game. Each Human Intelligence Task (HIT) consists of 10 games (each game corresponds to one pool) and we find that overall 76.7% of users who started a HIT completed it *i.e.* played all 10 games. We note that incomplete game data was discarded and does not contribute to the analysis presented in subsequent sections.

We published HITs until 28 games with both $\textsc{Alice}_{\texttt{SL}}$ and $\textsc{Alice}_{\texttt{RL}}$ were completed. This results in a total of 560 games split between the agents, with each game consisting of 9 rounds of dialog and 10 rounds of guessing. Workers are paid a base pay of \$5 per HIT (~\$10/hour).

To incentivize workers to try their best at guessing the secret image, workers are paid a two-part bonus – (1) based on the number of times their best guess matched the true secret image after each round (up to \$1 per HIT), and (2) based on the rank of the true secret image in their final sorting at the end of dialog (up to \$2 per HIT).

This final ranking explicitly captures the workers' mental model of the secret image (unlike the per-round, best-guess estimates), and is closer to the overall purpose of the game (identifying the secret image at the end of the dialog). As such, this final sorting is given a higher potential bonus.

A.4.3 Evaluation

Since the game is structured as a retrieval task, we evaluate the human-AI collaborative performance using standard retrieval metrics. Note that the successive selection of images by H at the end of the dialog tells us the rank of the true secret image in a sorting of the image pool based on H's mental model. For example, if H makes 4 guesses before correctly selecting the secret image, then H's mental model ranked the secret image 5th within the pool.

To evaluate human-AI collaboration, we use the following metrics: (1) Mean Rank (MR), which is the mean rank of the secret image (*i.e.* number of guesses it takes to identify the secret image). Lower values indicate better performance. (2) Mean Reciprocal Rank (MRR), which is the mean of the reciprocal of the rank of the secret image. MRR penalizes differences in lower ranks (e.g., between 1 and 2) greater than those in higher ranks (e.g., between 19 and 20). Higher values indicate better performance.

At the end of each round, H makes their best guess of the secret image. To get a coarse estimate of the rank of the secret image in each round, we sort the image pool based on distance in fc7 embedding space from H's best guess. This can be used to assess accuracy of H's mental model of the secret image after each round of dialog (e.g., Fig. A.4b).

A.5 Infrastructure

We briefly outline the backend architecture of GuessWhich in this section. Unlike most human-labeling tasks that are one-way and static in nature (i.e., only involving a human labeling static data), evaluating AI agents via our game requires live interaction between

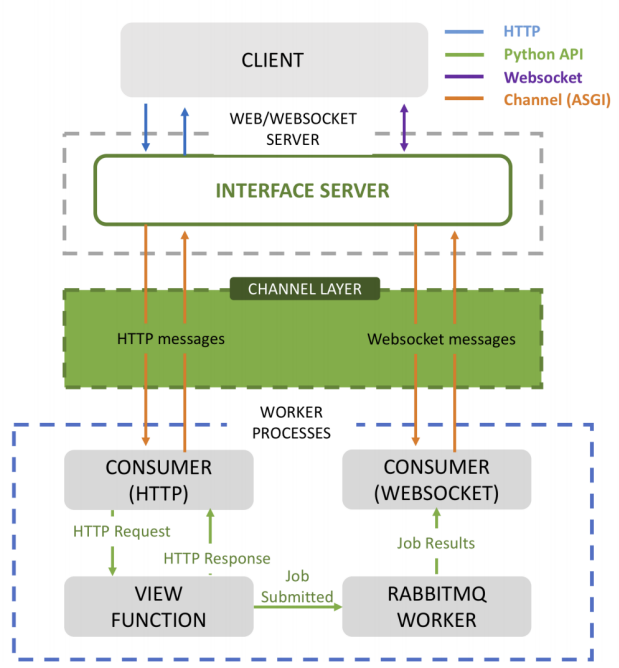
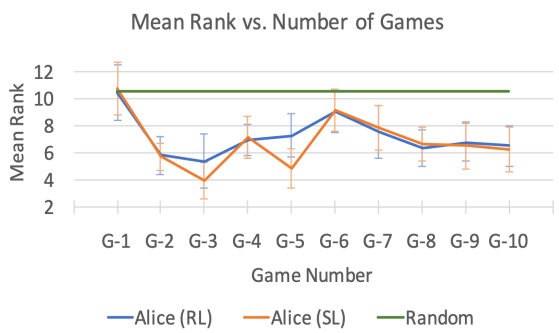


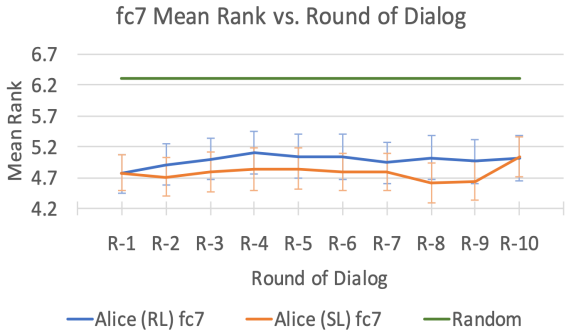
Figure A.3: We outline the backend architecture of our implementation of GuessWhich. Since GuessWhich requires a live interaction between the human and the AI, we design a workflow that can handle multiple queues and can quickly pair a human with an AI agent.

the AI agent and the human. We develop a robust workflow that can maintain a queue of workers and pair them up in real-time with an AI agent.

We deploy ALICE$_{\text{SL}}$ and ALICE$_{\text{RL}}$ on an AWS EC2 [56] GPU instance. We use Django (a Model-View-Controller web framework written in Python) which helps in monitoring HITs in real-time. We use [57], an open source message broker, to queue inference jobs that generate dialog responses from the model. Our backend is asynchronously connected to the client browser via websockets such that whenever an inference job is completed, a websocket polls the AI response and delivers it to the human in real-time. We store and fetch data efficiently to and from a PostgreSQL database. Fig. A.3 shows a schematic diagram of the backend architecture. Our complete backend infrastructure and code will be made publicly available for others to easily make use of our human-AI game interface.



(a) ALICE$_{SL}$ and ALICE$_{RL}$ perform about the same for most games and outperform a baseline model that makes a string of random guesses at the end of each game.



(b) ALICE$_{SL}$ and ALICE$_{RL}$ perform about the same, and clearly outperform a baseline model that randomly chooses an image. As described in Sec. A.4.3, this is only a coarse estimate of the rank of the secret image after each round of dialog.

Figure A.4: Mean rank (MR) of secret image across (a) number of games and (b) rounds of dialog. Lower is better. Error bars are 95% confidence intervals from 1000 bootstrap samples.

A.6 Results

A.6.1 ALICE$_{SL}$ vs. ALICE$_{RL}$

We compare the performance of the two agents ALICE$_{SL}$ and ALICE$_{RL}$ in the GuessWhich game. These bots are state-of-the-art visual dialog agents with respect to emulating human responses and generating visually discriminative responses in AI-AI dialog. [23] evaluate these agents against strong baselines and report AI-AI team results that are significantly better than chance on a pool of ~10k images (rank ~1000 for SL, rank ~500 for RL). In addition to evaluating them in the context of human-AI teams we also report QBOT-ALICE team performances for reference.

In Table A.1, we compare the performances of human-ALICE$_{SL}$ and human-ALICE$_{RL}$ teams according to Mean Rank (MR) and Mean Reciprocal Rank (MRR) of the secret image based on the guesses H makes at the end of dialog. We observe that at the end of each game (9 rounds of dialog), human subjects correctly guessed the secret image on their 6.86th attempt (Mean Rank) when ALICE$_{SL}$ was their teammate. With ALICE$_{RL}$ as their teammate, the average number of guesses required was 7.19. We also observe that ALICE$_{RL}$

Table A.1: Performance of Human-ALICE teams with ALICE_{SL} and ALICE_{RL} measured by MR (lower is better) and MRR (higher is better). Error bars are 95% CIs from 1000 bootstrap samples. Unlike (Das et al., 2017b), we find no significant difference between ALICE_{SL} and ALICE_{RL}.

Team	**MR**	**MRR**
Human-ALICE_{SL}	6.86 ± 0.53	0.27 ± 0.03
Human-ALICE_{RL}	7.19 ± 0.55	0.25 ± 0.03

Table A.2: Performance of Human-ALICE and QBOT-ALICE teams measured by MR (lower is better). We observe that AI-AI teams outperform human-AI teams.

Team	**Alice$_{\text{SL}}$**	**Alice$_{\text{RL}}$**
Human	6.9	7.2
QBOT (SL)	5.6	5.3
QBOT (RL)	4.7	4.7

outperforms ALICE_{SL} on the MRR metric. On both metrics, however, the differences are within the standard error margins (reported in the table) and not statistically significant. As we collected additional data, the error margins became smaller but the means also became closer. This interesting finding stands in stark contrast to the results reported by [23], where ALICE_{RL} was found to be significantly more accurate than ALICE_{SL} when evaluated in an AI-AI team. Our results suggest that the improvements of RL over SL (in AI-AI teams) do not seem to translate to when the agents are paired with a human in a similar setting.

MR with varying number of games. In Fig. A.4a, we plot the mean rank (MR) of the secret image across different games. We see that the human-ALICE team performs about the same for both ALICE_{SL} and ALICE_{RL} except Game 5, where ALICE_{SL} seems to marginally outperform ALICE_{RL}. We compare the performance of these teams against a baseline model that makes a string of random guesses at the end of the game. The human-ALICE teams outperforms this random baseline with a relative improvement of about 25%.

AI-Alice teams versus human-Alice teams. In Table A.2, we compare team performances by pairing three kinds of questioners – human, QBOT (SL) and QBOT (RL) with ALICE_{SL} and ALICE_{RL} (6 teams in total) to gain insights about how the questioner and ALICE influence team performances. Interestingly, we observe that AI-ALICE teams outperform human-ALICE teams. On average, a QBOT (SL)-ALICE_{SL} team takes about 5.6 guesses to arrive at the correct secret image (as opposed to 6.86 guesses for a human-ALICE_{SL} team). Similarly, a QBOT (RL)-ALICE_{RL} team takes 4.7 guesses as opposed to a human-ALICE_{RL} team which takes 7.19 guesses. When we compare AI-AI teams (see Row 2 and 3) under different settings, we observe that teams having QBOT (RL) as the questioner outperform those with QBOT (SL). Qualitatively, we found that QBOT (SL) tends to ask repeating questions in a dialog and that questions from QBOT (RL) tend to be more visually grounded compared to QBOT (SL). Also, note that among the four teams ALICE does not seem to affect performance across SL and RL.

Since we observe that QBOT (RL) tends to be a better questioner on average compared to QBOT (SL), as future work, it will be interesting to explore a setting where we evaluate QBOT via a similar game with the human playing the role of answerer in a QBOT-human team.

MR with varying rounds of dialog. Fig. A.4b shows a coarse estimate of the mean rank of the secret image across rounds of a dialog, averaged across games and workers. As explained in Sec. A.4.3, image ranks are computed via distance in embedding space from the guessed image (and hence, are only an estimate). We see that the human-ALICE team performs about the same for both ALICE_{SL} and ALICE_{RL} across rounds of dialog in a game. When compared with a baseline agent that makes random guesses after every round of dialog, the human-ALICE team clearly performs better.

Statistical tests. Observe that on both the metrics (MR and MRR), the differences between

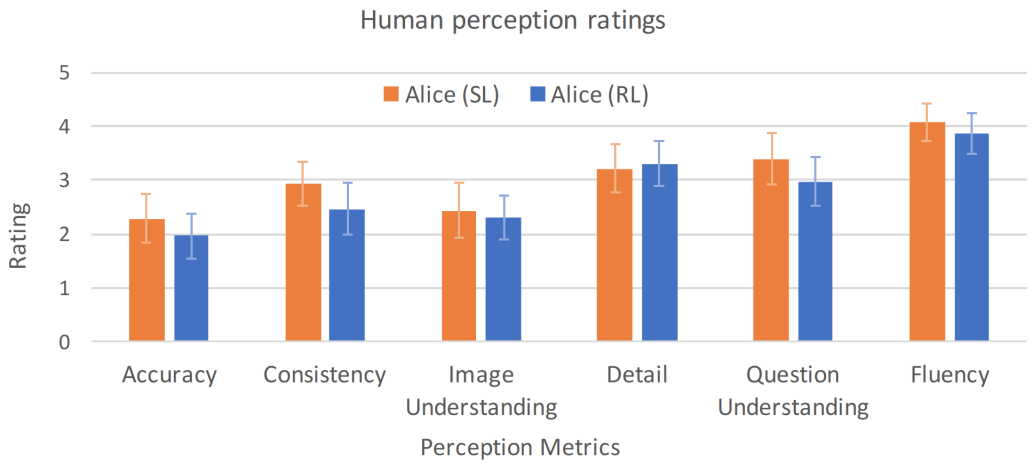


Figure A.5: Worker ratings for ALICE$_{SL}$ and ALICE$_{RL}$ on 6 metrics. Higher is better. Error bars are 95% confidence intervals from 1000 bootstrap samples. Humans perceive no significant differences between ALICE$_{SL}$ and ALICE$_{RL}$ across the 6 feedback metrics.

performances of ALICE$_{SL}$ and ALICE$_{RL}$ are within error margins. Since both standard error and bootstrap based 95% confidence intervals overlap significantly, we ran further statistical tests. We find no significant difference between the mean ranks of ALICE$_{SL}$ and ALICE$_{RL}$ under a Mann-Whitney U test ($p = 0.44$).

A.6.2 Human perception of AI teammate

At the end of each HIT, we asked workers for feedback on ALICE. Specifically, we asked workers to rate ALICE on a 5-point scale (where 1=Strongly disagree, 5=Strongly agree), along 6 dimensions. As shown in Fig. A.5, ALICE was rated on – how accurate they thought it was (accuracy), how consistent its answers were with its previous answers (consistency), how well it understood the secret image (image understanding), how detailed its answers were (detail), how well it seemed to understand their questions (question understanding) and how fluent its answers were (fluency).

We see in Fig. A.5 that humans perceive both ALICE$_{SL}$ and ALICE$_{RL}$ as comparable in terms of all metrics. The small differences in perception are not statistically significant.

A.6.3 Questioning Strategies

Fig. A.6 shows the distribution of questions that human subjects ask ALICE in GuessWhich. Akin to the format of the human-human GuessWhat game, we observe that binary (yes/no) questions are overwhelmingly the most common question type, for instance, "Is there/the/he ...?" (region shaded yellow in the figure), "Are there ...?" (region shaded red), etc. The next most frequent question is "What color ...?". These questions may be those that help the human discriminate the secret image the best. It could also be that humans are attempting to play to the perceived strengths of ALICE. As people play multiple games with ALICE, it is possible that they discover ALICE's strengths and learn to ask questions that play to its strengths. Another common question type is counting questions, such as "How many ...?". Interestingly, some workers adopt the strategy of querying ALICE with a single word (e.g., nouns such as "people", "pictures", etc.) or a phrase (e.g., "no people", "any cars", etc.). This strategy, while minimizing human effort, does not appear to change ALICE's performance. Fig. A.7 shows a game played by two different subjects.

A.7 Challenges

There exist several challenges that are unique to human computation in the context of evaluating human-AI teams, for instance, making our games engaging while still ensuring fair and accurate evaluation. In this section, we briefly discuss some of the challenges we faced and our solutions to them.

Knowledge Leak. It has been shown that work division in crowdsourcing tasks follows a Pareto principle [58], as a small fraction of workers usually complete a majority of the work. In the context of evaluating an AI based on performance of a human-AI team, this poses a challenge.

Recently, [59] showed that human subjects can predict the responses of an AI more accurately with higher familiarity with the AI. That is, a human's knowledge gained from

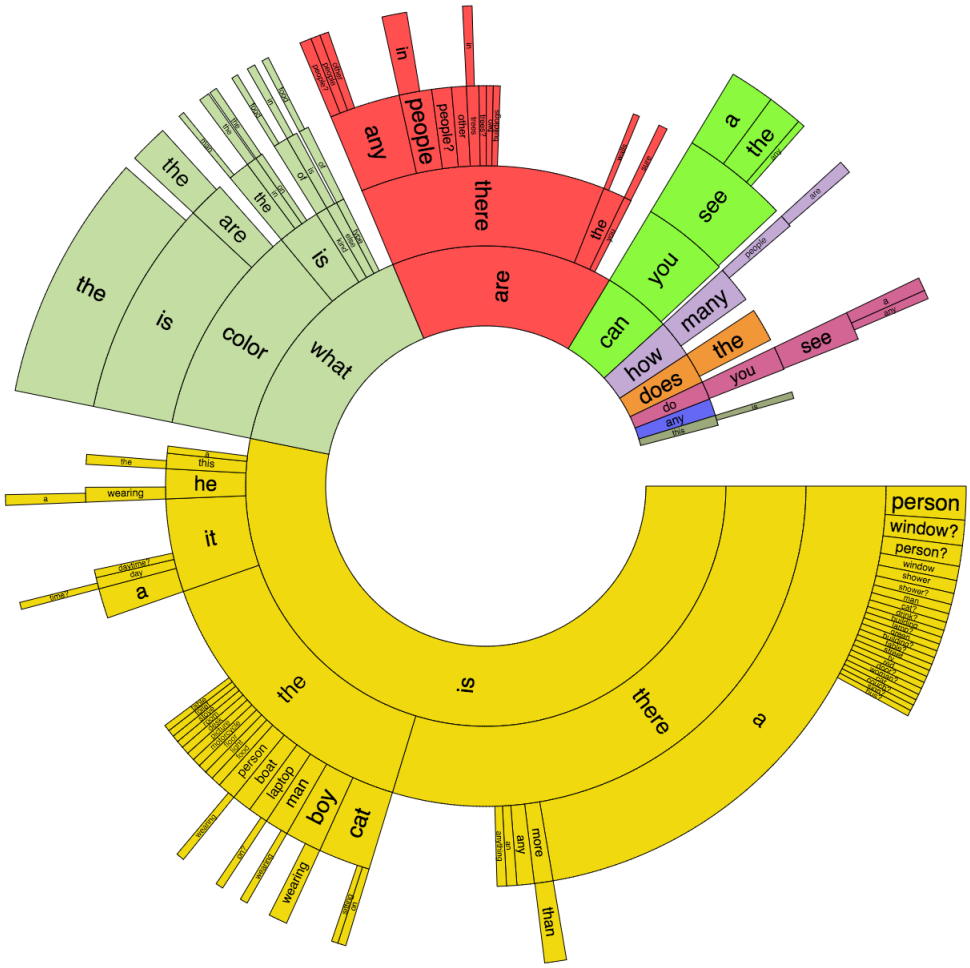


Figure A.6: Distribution of first n-grams for questions asked to Alice. Word ordering starts from the center and radiates outwards. Arc length is proportional to the number of questions containing the word. The most common question-types are binary – followed by 'What color..' questions.

familiarity with their AI teammate, can bias the performance of the human-AI team – knowledge from previous tasks might leak to later tasks. To prevent a biased evaluation of team performance due to human subjects who have differing familiarity with Alice, every person only plays a fixed number of games (10) with Alice. Thus, a human subject can only accept one task on AMT, which involves playing 10 games. The downside to this is that our ability to conduct a fair evaluation of an AI in an interactive, game-like setting is constrained by the number of unique workers who accept our tasks.

Engagement vs. Fairness. In order to improve user-engagement while playing our games,

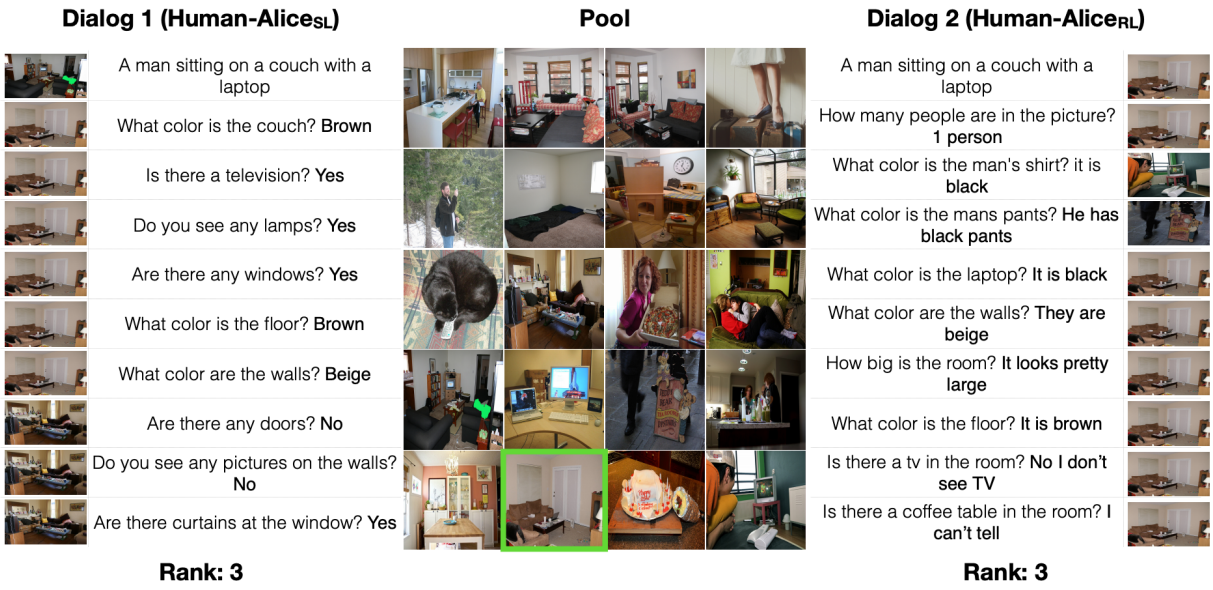


Figure A.7: We contrast two games played by different workers with ALICE$_{SL}$ and ALICE$_{RL}$ on the same pool (secret image outlined in green). In both cases, the workers are able to find the secret image within three guesses. It is also interesting to note how the answers provided by ALICE are different in the two cases.

we offer subjects performance-based incentives that are tied to the success of the human-AI team. There is one potential issue with this however. Owing to the inherent complexity of the visual dialog task, ALICE tends to be inaccurate at times. This increases both the difficulty and unpredictability of the game, as it tends to be more accurate for certain types of questions compared to others. We observe that this often leads to unsuccessful game-plays, sometimes due to errors accumulating from successive incorrect responses from ALICE to questions from the human. In a few other cases, the human is misled by ALICE by a single wrong answer or by the seed caption that tends to be inaccurate at times. While we would like to keep subjects engaged in the game to the best extent possible by providing performance-based incentives, issuing a performance bonus that depends on both the human and ALICE (who is imperfect), can be dissatisfying. To be fair to the subjects performing the task while still rewarding good performance, we split our overall budget for each HIT into a suitable fraction between the base pay (majority), and the performance bonus.

A.8 Conclusion

In contrast to the common practice of measuring AI progress in isolation, our work proposes benchmarking AI agents via interactive downstream tasks (cooperative games) performed by human-AI teams. In particular, we evaluate visual conversational agents in the context of human-AI teams. We design a cooperative game – GuessWhich – that involves a human engaging in a dialog with an answerer-bot (ALICE) to identify a secret image known to ALICE but unknown to the human from a pool of images. At the end of the dialog, the human is asked to pick out the secret image from the image pool by making successive guesses. We find that ALICE_{RL} (fine-tuned with reinforcement learning) that has been found to be more accurate in AI literature than it's supervised learning counterpart when evaluated via a questioner bot (QBOT)-ALICE team, is not more accurate when evaluated via a human-ALICE team. This suggests that there is a disconnect between between benchmarking of AI in isolation versus in the context of human-AI interaction. An interesting direction of future work could be to evaluate QBOT via QBOT-human teams.

APPENDIX B
DO EXPLANATION MODALITIES MAKE VQA MODELS MORE PREDICTABLE TO A HUMAN?

A rich line of research attempts to make deep neural networks more transparent by generating human-interpretable 'explanations' of their decision process, especially for interactive tasks like Visual Question Answering (VQA). In this work, we analyze if existing explanations indeed make a VQA model – its responses as well as failures – more predictable to a human. Surprisingly, we find that they do not. On the other hand, we find that human-in-the-loop approaches that treat the model as a black-box do.

B.1 Introduction

As technology progresses, we are increasingly collaborating with AI agents in *interactive* scenarios where humans and AI work together as a team, e.g., in AI-assisted diagnosis, autonomous driving, etc. Thus far, AI research has typically only focused on the AI in such an interaction – for it to be more accurate, be more human-like, understand our intentions, beliefs, contexts, and mental states.

In this work, we argue that for human-AI interactions to be more effective, humans must also understand the AI's beliefs, knowledge, and quirks.

Many recent works generate human-interpretable 'explanations' regarding a model's decisions. These are usually evaluated offline based on whether human judges found them to be 'good' or to improve trust in the model. However, their contribution in an interactive setting remains unclear. In this work, we evaluate the role of explanations towards making a model predictable to a human.

We consider an AI trained to perform the multi-modal task of Visual Question Answering (VQA) [60, 61], i.e., answering free-form natural language questions about images. VQA

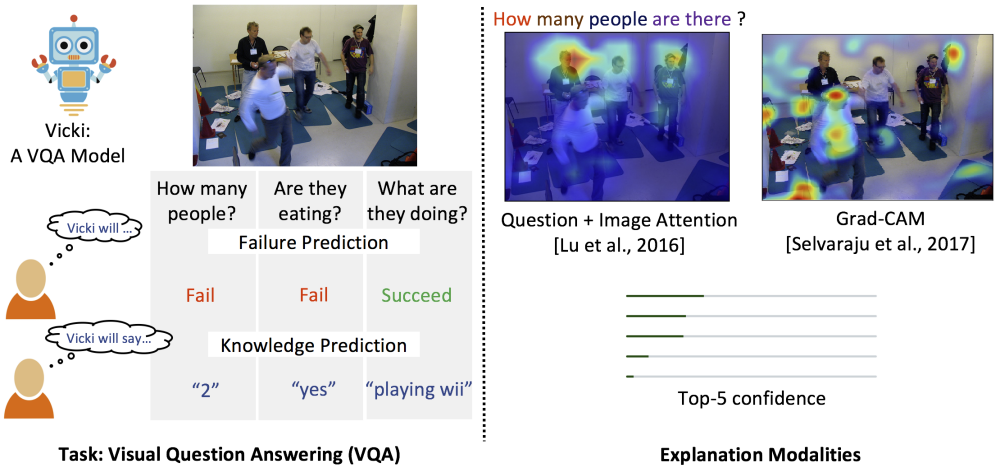


Figure B.1: We evaluate the extent to which explanation modalities (right) and familiarization with a VQA model help humans predict its behavior – its responses, successes, and failures (left).

is applicable to scenarios where humans actively elicit information from visual data, and naturally lends itself to human-AI interactions. We consider two tasks that demonstrate the degree to which a human understands their AI teammate (we call Vicki) – Failure Prediction (FP) and Knowledge Prediction (KP). In FP, we ask subjects on Amazon Mechanical Turk to predict if Vicki will correctly answer a given question about an image. In KP, subjects predict Vicki's exact response.

We aid humans in forming a mental model of Vicki by (1) familiarizing them with its behavior in a 'training' phase and (2) exposing them to its internal states via various explanation modalities. We then measure their FP and KP performance.

Our key findings are that (1) humans are indeed capable of predicting successes, failures, and outputs of the VQA model better than chance, (2) explicitly training humans to familiarize themselves with the model improves their performance, and (3) existing explanation modalities do not enhance human performance.

B.2 Related Work

Explanations in deep neural networks. Several works generate explanations based on internal states of a decision process[62, 63], while others generate justifications that are

consistent with model outputs [64, 65]. Another popular form of providing explanations is to visualize regions in the input that contribute to a decision – either by explicitly attending to relevant input regions [66, 67], or exposing implicit attention for predictions [68, 69].

Evaluating explanations. Several works evaluate the role of explanations in developing trust with users [70, 64] or helping them achieve an end goal [71, 72]. Our work, however, investigates the role of machine-generated explanations in improving the *predictability* of a VQA model.

Failure prediction. While [73] and [74] predict failures of a model using simpler statistical models, we explicitly train a person to do this.

Legibility. [75] describe the intent-expressiveness of a robot as its trajectory being expressive of its goal. Analogously, we evaluate if explanations of the intermediate states of a VQA model are expressive of its output.

Humans adapting to technology. [76] and [77] observe humans' strategies while adapting to the limited capabilities of an AI in interactive language games. In our work we explicitly measure to what extent humans can form an accurate model of an AI, and the role of familiarization and explanations.

B.3 Setup

Agent. We use the VQA model by [78] as our AI agent (that we call Vicki). The model processes the question at multiple levels of granularity (words, phrases, entire question) and at each level, has explicit attention mechanisms on both the image and the question[1]. It is trained on the train split of the VQA-1.0 dataset [61]. Given an image and a question about the image, it outputs a probability distribution over 1000 answers. Importantly, the model's image and question attention maps provide access to its 'internal states' while making a prediction.

Vicky is *quirky* at times, i.e., has biases, albeit in a predictable way. [79] outlines several

[1]We use question-level attention maps in our experiments.

such quirks. For instance, Vicki has a limited capability to understand the image – when asked the color of a small object in the scene, say a soda can, it may simply respond with the most dominant color in the scene. Indeed, it may answer similarly even if no soda can is present, i.e. if the question is irrelevant.

Further, Vicki has a limited capability to understand free-form natural language, and in many cases, answers questions based only on the first few words of the question. It is also generally poor at answering questions requiring "common sense" reasoning. Moreover, being a discriminative model, Vicki has a limited vocabulary (1k) of answers. Additionally, the VQA 1.0 dataset contains label biases; therefore, the model is very likely to answer "white' to a "what color" question [80].

To get a sense for this, see Fig. B.2 which depicts a clear pattern. In top-left, even when there is no grass, Vicki tends to latch on to one of the dominant colors in the image. For top-right, even when there are no people in the image, it seems to respond with what people could *plausibly* do in the scene if they were present.

In this work, we measure to what extent lay people can pick up on these quirks by interacting with the agent, and whether existing explanation modalities help do so.

Tasks: Failure Prediction (FP). Given an image and a question about the image, we measure how well a person can predict if Vicki will successfully answer the question. A person can presumably predict the failure modes of Vicki well if they have a good sense of its strengths and weaknesses.

Knowledge Prediction (KP). In this task, we aim to obtain a fine-grained measure of a person's understanding of Vicki's behavior. Given a QI–pair, a subject guesses Vicki's exact response from a set of its output labels.

Snapshots of our interfaces can be seen in Fig. B.3.

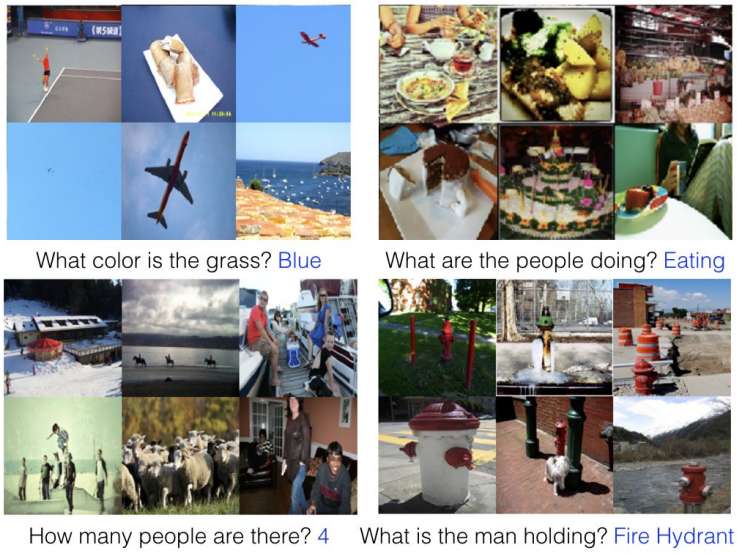


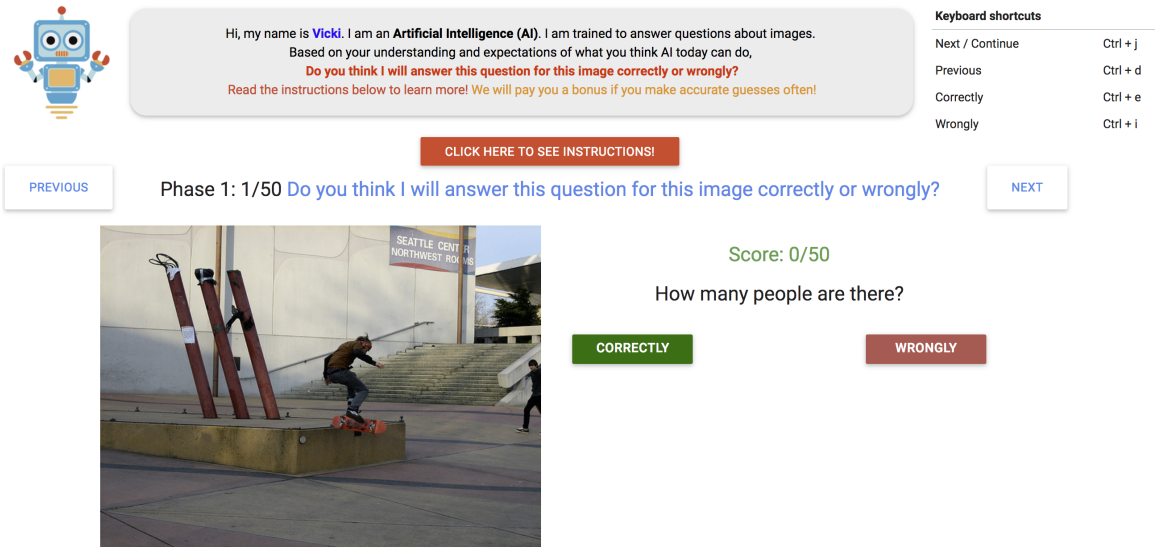
Figure B.2: These montages highlight some of Vicki's quirks. For a given question, Vicki has the same response to each image in a montage. Common visual patterns (that Vicki presumably picks up on) within each montage are evident.

B.4 Experimental Setup

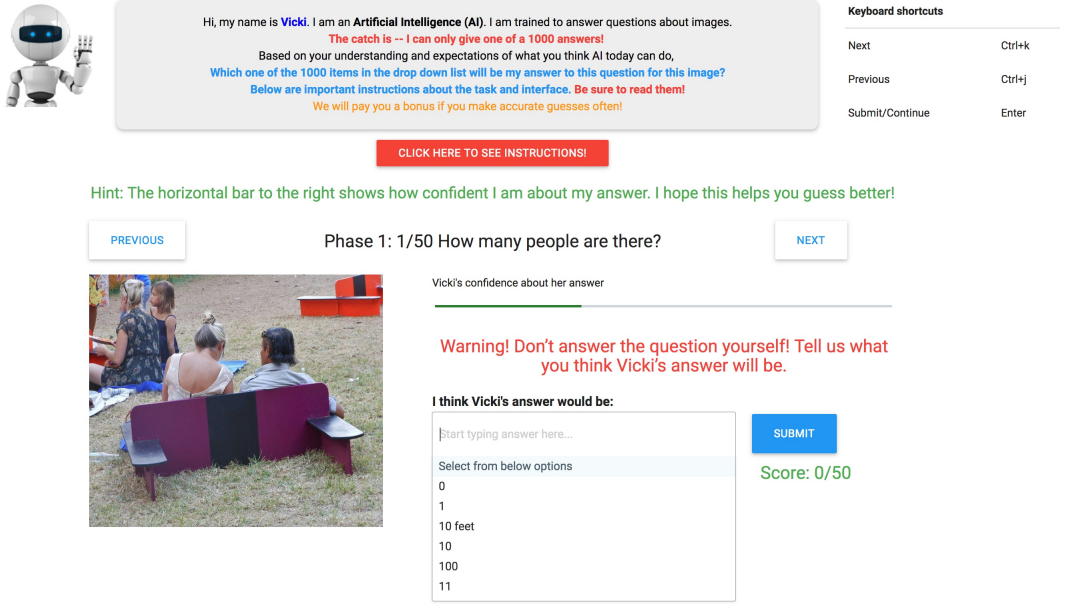
In this section we investigate ways to make Vicki's behavior more predictable to a subject. We approach this by – providing instant feedback about Vicki's actual behavior on each QI pair once the subject responds, and exposing subjects to various explanation modalities that reveal Vicki's internal states before they respond.

Data. We identify a subset of questions in the VQA-1.0 [61] validation split that occur more than 100 times. We select 7 diverse questions[2] from this subset that are representative of the different types of questions (counting, yes/no, color, scene layout, activity, etc.) in the dataset. For each of the 7 questions, we sample a set of 100 images. For FP, the 100 images are random samples from the set of images on which the question was asked in VQA-1.0 val. For the KP task, these 100 images are random images from VQA-1.0 val. [81] found that randomly pairing an image with a question in the VQA-1.0 dataset results in about 79% of pairs being irrelevant. This combination of relevant and irrelevant QI-pairs allows us to test

[2]What kind of animal is this? What time is it? What are the people doing? Is it raining? What room is this? How many people are there? What color is the umbrella?



(a) The Failure Prediction (FP) interface.



(b) The Knowledge Prediction (KP) interface.

Figure B.3: (a) A person guesses if a VQA model (Vicki) will answer this question for this image correctly or wrongly. (b) A person guesses what Vicki's exact answer will be for this QI–pair.

subjects' ability to develop a robust understanding of Vicki's behavior across a wide variety of inputs.

Study setup. We conduct our studies on Amazon Mechanical Turk. Each task (HIT) com-

prises of 100 QI-pairs where for simplicity (for the subject), a single question is asked across all 100 images. The annotation task is broken down into a train and test phase of 50 QI-pairs each. Over all settings, 280 workers took part in our study (1 unique worker per HIT), resulting in 28k human responses. Subjects were paid an average of $3 base plus $0.44 performance bonus, per HIT.

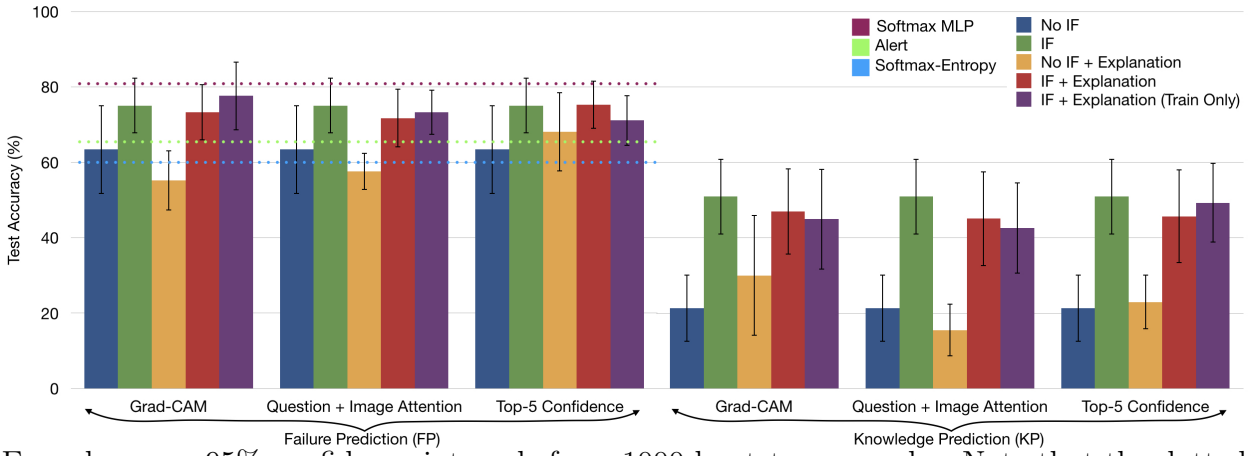
There are some challenges involved in scaling data-collection in this setting: (1) Due to the presence of separate train and test phases, our AMT tasks tend to be unusually long (mean HIT durations across the tasks of FP and KP = 10.11 ± 1.09 and 24.49 ± 1.85 min., respectively). Crucially, this also reduces the subject pool to only those willing to participate in long tasks. (2) Once a subject participates in a task, they cannot do another because their familiarity with Vicki would leak over. This constraint causes our analyses to require as many subjects as tasks. Since work division in crowdsourcing tasks follows a Pareto principle [58], this makes data collection very slow.

In light of these challenges, we focus on a small set of questions to systematically evaluate the role of training and exposure to Vicki's internal states.

B.4.1 Evaluating the role of familiarization

To familiarize subjects with Vicki, we provide them with instant feedback during the train phase. Immediately after a subject responds to a QI–pair, we show them whether Vicki actually answered the question correctly or not (in FP) or what Vicki's response was (in KP), along with a running score of how well they are doing. Once training is complete, no further feedback is provided and subjects are asked to make predictions for the test phase. At the end, they are shown their score and paid a bonus proportional to the score.

Failure Prediction. In FP, always guessing that Vicki answers 'correctly' results in 58.29% accuracy, while subjects do slightly better and achieve 62.66% accuracy, even without prior familiarity with Vicki (No Instant Feedback (IF)). Further, we find that subjects that receive training via instant feedback (IF) achieve 13.09% higher mean accuracies than those who do



Error bars are 95% confidence intervals from 1000 bootstrap samples. Note that the dotted lines are various machine approaches applied to FP.

Error bars are 95% confidence intervals from 1000 bootstrap samples. Note that the dotted lines are various machine approaches applied to FP.

Figure B.4: Average performance across subjects for Failure Prediction and Knowledge Prediction, across different settings: with or without (1) Instant feedback (IF) in the train phase, and (2) an explanation modality. xplanation modalities are shown in both train and test phases unless stated otherwise.
Error bars are 95% confidence intervals from 1000 bootstrap samples. Note that the dotted lines are various machine approaches applied to FP.

not (see Fig B.4.1; IF vs No IF for FP (left)).

Knowledge Prediction. In KP, answering each question with Vicki's most popular answer overall ('no') would lead to an accuracy of 13.4%. Additionally, answering each question with its most popular answer *for that question* leads to an accuracy of 31.43%. Interestingly, subjects who are unfamiliar with Vicki (No IF) achieve 21.27% accuracy – better than the most popular answer overall, but worse than the question-specific prior over its answers. The latter is understandable as subjects unfamiliar with Vicki do not know which of its 1000 possible answers the model is most likely to predict for each question.

We find that mean performance in KP with IF is 51.11%, 29.84% higher than KP without IF (see Fig B.4.1; IF vs No IF for KP (right)). It is apparent that just from a few (50) training examples, subjects succeed in building a mental model of Vicki's behavior that generalizes to new images. Additionally, the 29.84% improvement over No IF for KP is significantly larger than that for FP (13.09%). This is understandable because a priori (No IF), KP is a much

harder task as compared to FP due to the increased space of possible subject responses given a QI-pair, and the combination of relevant and irrelevant QI-pairs in the test phase.

Questions such as 'Is it raining?' have strong language priors – to these Vicki often defaults to the most popular answer ('no'), irrespective of image. On such questions, subjects perform considerably better in KP once they develop a sense for Vicki's inherent biases via instant feedback. For open-ended questions like 'What time is it?', feedback helps subjects (1) narrow down the 1000 potential options to the subset that Vicki typically answers with – in this case time periods such as 'daytime' rather than actual clock times and (2) identify correlations between visual patterns and Vicki's answer. In other cases like 'How many people are in the image?' the space of possible answers is clear a priori, but after IF subjects realize that Vicki is bad at detailed counting and bases its predictions on coarse signals of the scene layout.

B.4.2 Evaluating the role of explanations

In this setting, we show subjects an image, a question, and one of the explanation modalities described below. We experiment with 3 qualitatively different modalities (see Fig.B.1, right): **Confidence of top-5 predictions.** We show subjects Vicki's confidence in its top-5 answer predictions from its vocabulary as a bar plot (of course, we do not show the actual top-5 predictions). **Attention maps.** Along with the image we show subjects the spatial attention map over the image and words of the question which indicate the regions that Vicki is looking at and listening to, respectively. **Grad-CAM.** We use the CNN visualization technique by [68] [68], using the (implicit) attention maps corresponding to Vicki's most confident answer.

Automatic approaches. We also evaluate automatic approaches to detect Vicki's failure from its internal states. We find that both, a decision stump on Vicki's confidence in its top answer, and on the entropy of its softmax output, result in an FP accuracy of 60% on our test set. A Multi-layer Perceptron (MLP) trained on Vicki's output 1000-way softmax to predict success vs failure, achieves an FP accuracy of 81%. Training on just top-5 softmax

outputs achieves an FP accuracy of 61.43%.

Training an MLP which takes as input question features (average word2vec embeddings [82] of words in the question) concatenated with image features (fc7 from VGG-19) to predict success vs failure (which we call ALERT following [74]) achieves an FP accuracy of 65%. Training an MLP on identical question features as above but concatenated with Grad-CAM saliency maps leads to FP accuracy of 73.14%. [3] Note that we only report machine results to put human accuracies in perspective. We do not draw any inferences about the relative capabilities of both.

Results. Average performance of subjects in the test phases of FP and KP, for different experimental settings are summarized in Fig. B.4.1. In the first setting, we show subjects an explanation modality with instant feedback (IF+Explanation). For reference, also see performance of subjects provided with IF and no explanation modality (IF).

We observe that on both FP and KP, subjects who received an explanation along with IF show no statistically significant difference in performance compared to those who did not. We see in Fig. B.4.1, that both bootstrap based standard error (95% confidence intervals) overlap significantly.

Seeing that explanations in addition to IF does not outperform an IF baseline, we next measure whether explanations help a user not already familiar with Vicki via IF. That is, we evaluate if explanations help against a No IF baseline by providing an explanation only in the *test* phase, and no IF (see Fig B.4.1; No IF + Explanation). Additionally, we also experiment with providing IF and an explanation *only* during the train phase (see Fig B.4.1; IF + Explanation (Train Only)), to measure whether access to internal states during training can help subjects build better intuitions for model behavior without needing access to internal states at test time. In both settings however, we observe no statistically significant difference in performance over the No IF and IF baselines, respectively. [4]

[3] These methods are trained on 66% of VQA-1.0 val. The remaining data is used for validation.

[4] When piloting the tasks ourselves, we found it easy to 'overfit' to the explanations and hallucinate patterns.

B.5 Conclusion

As technology progresses, human-AI teams are inevitable. We argue that for these teams to be more effective, we should also be pursuing research directions to help humans understand the strengths, weaknesses, quirks, and tendencies of AI. We instantiate these ideas in the domain of Visual Question Answering (VQA), by proposing two tasks that help measure how well a human 'understands' a VQA model (we call Vicki) – Failure Prediction (FP) and Knowledge Prediction (KP). We find that lay people indeed get better at predicting Vicki's behavior using just a few 'training' examples, but surprisingly, existing popular explanation modalities do not help make its failures or responses more predictable. While previous works have typically assessed their interpretability or their role in improving human trust, our preliminary hypothesis is that these modalities may not yet help performance of human-AI teams in a goal-driven setting. Clearly, much work remains to be done in developing improved explanation modalities that can improve human-AI teams.

Future work involves closing the loop and evaluating the extent to which improved human performance at FP and KP translates to improved success of human-AI teams at accomplishing a shared goal. Co-operative human-AI games may be a natural fit for such an evaluation.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *NIPS*, 2012.

[2] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, "VQA: Visual Question Answering," in *International Conference on Computer Vision (ICCV)*, 2015.

[3] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[4] OpenAI, *Openai five*, `https://blog.openai.com/openai-five/`, 2018.

[5] *Amazon Mechanical Turk (AMT)*, Website - .

[6] P. Chattopadhyay, D. Yadav, V. Prabhu, A. Chandrasekaran, A. Das, S. Lee, D. Batra, and D. Parikh, "Evaluating visual conversational agents via cooperative human-ai games," in *Proceedings of the Fifth AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*, 2017.

[7] E. Kolve, R. Mottaghi, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi, "Ai2-thor: An interactive 3d environment for visual ai," *ArXiv preprint arXiv:1712.05474*, 2017.

[8] A. Das, S. Datta, G. Gkioxari, S. Lee, D. Parikh, and D. Batra, "Embodied Question Answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[9] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*, Springer, 2014, pp. 740–755.

[10] A. Das, S. Kottur, K. Gupta, A. Singh, D. Yadav, J. M. Moura, D. Parikh, and D. Batra, "Visual Dialog," in *CVPR*, 2017.

[11] *Kaggle*, Website - `https://kaggle.com/`.

[12] *CodaLab*, Website - `https://competitions.codalab.org/`.

[13] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *CoRR*, vol. abs/1207.4708, 2012.

[14] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," *CoRR*, vol. abs/1604.06778, 2016.

[15] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *CoRR*, vol. abs/1606.01540, 2016.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[18] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[19] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Computer Vision (ICCV), 2017 IEEE International Conference on*, IEEE, 2017, pp. 2980–2988.

[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[21] B. Dai and D. Lin, "Towards diverse and natural image descriptions via a conditional gan,"

[22] D. Li, X. He, Q. Huang, M.-T. Sun, and L. Zhang, "Generating diverse and accurate visual captions by comparative adversarial learning," *ArXiv preprint arXiv:1804.00861*, 2018.

[23] A. Das, S. Kottur, J. M. Moura, S. Lee, and D. Batra, "Learning Cooperative Visual Dialog Agents with Deep Reinforcement Learning," in *ICCV*, 2017.

[24] *Docker*, Website - `https://www.docker.com/`.

[25] *Amazon Elastic Container Service (ECS)*, Website - `https://aws.amazon.com/ecs/`.

[26] *Django: The web framework for perfectionists with deadlines.* Website - `https://www.djangoproject.com/`.

[27] *Node.js*, Website - `https://nodejs.org/`.

[28] *Socket.IO*, Website - `http://socket.io/`.

[29] *Amazon Simple Queue Service*, Website - `https://aws.amazon.com/sqs/`.

[30] C.-W. Liu, R. Lowe, I. V. Serban, M. Noseworthy, L. Charlin, and J. Pineau, "How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation," *ArXiv preprint arXiv:1603.08023*, 2016.

[31] H. Agrawal, C. S. Mathialagan, Y. Goyal, N. Chavali, P. Banik, A. Mohapatra, A. Osman, and D. Batra, "Cloudcv: Large scale distributed computer vision as a cloud service," *CoRR*, vol. abs/1506.04130, 2015. arXiv: `1506.04130`.

[32] A. Ray, G. Christie, M. Bansal, D. Batra, and D. Parikh, "Question Relevance in VQA: Identifying Non-Visual And False-Premise Questions," in *EMNLP*, 2016.

[33] A. Agrawal, D. Batra, and D. Parikh, "Analyzing the behavior of visual question answering models," *CoRR*, vol. abs/1606.07356, 2016. arXiv: `1606.07356`.

[34] H. de Vries, F. Strub, S. Chandar, O. Pietquin, H. Larochelle, and A. Courville, "GuessWhat?! visual object discovery through multi-modal dialogue," in *CVPR*, 2017.

[35] F. Strub, H. de Vries, J. Mary, B. Piot, A. C. Courville, and O. Pietquin, "End-to-end optimization of goal-driven and visually grounded dialogue systems," *ArXiv preprint arXiv:1703.05423*, 2017.

[36] L. Von Ahn and L. Dabbish, "Designing games with a purpose," *Communications of the ACM*, vol. 51, no. 8, pp. 58–67, 2008.

[37] L. von Ahn and L. Dabbish, "Labeling images with a computer game," in *CHI*, 2004.

[38] E. L. Law, L. Von Ahn, R. B. Dannenberg, and M. Crawford, "Tagatune: A game for music and sound annotation.," in *ISMIR*, vol. 3, 2007, p. 2.

[39] P. Michelucci, "Handbook of human computation," in *Springer*, 2013.

[40] S. Jain and D. C. Parkes, "A game-theoretic analysis of the esp game," *ACM Trans. Econ. Comput.*, vol. 1, no. 1, 3:1–3:35, Jan. 2013.

[41] M. Krause and J. Smeddinck, "Human computation games: A survey," in *Signal Processing Conference, 2011 19th European*, IEEE, 2011, pp. 754–758.

[42] L. Von Ahn, R. Liu, and M. Blum, "Peekaboom: A game for locating objects in images," in *Proceedings of the SIGCHI conference on Human Factors in computing systems*, ACM, 2006, pp. 55–64.

[43] E. Law and L. Von Ahn, "Input-agreement: A new mechanism for collecting data using human computation games," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 2009, pp. 1197–1206.

[44] S. Kazemzadeh, V. Ordonez, M. Matten, and T. L. Berg, "ReferItGame: Referring to Objects in Photographs of Natural Scenes," in *EMNLP*, 2014.

[45] N. Diakopoulos, K. Luther, and I. Essa, "Audio puzzler: Piecing together time-stamped speech transcripts with a puzzle game," in *Proceedings of the 16th ACM international conference on Multimedia*, ACM, 2008, pp. 865–868.

[46] H. Aras, M. Krause, A. Haller, and R. Malaka, "Webpardy: Harvesting qa by hc," in *Proceedings of the ACM SIGKDD Workshop on Human Computation*, ACM, 2010, pp. 49–52.

[47] J. Chamberlain, M. Poesio, and U. Kruschwitz, "Phrase detectives: A web-based collaborative annotation game," in *Proceedings of the International Conference on Semantic Systems (I-Semantics' 08)*, 2008, pp. 42–49.

[48] T. Paek, "Empirical methods for evaluating dialog systems," in *Proceedings of the workshop on Evaluation for Language and Dialogue Systems-Volume 9*, 2001.

[49] C.-W. Liu, R. Lowe, I. V. Serban, M. Noseworthy, L. Charlin, and J. Pineau, "How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation," in *EMNLP*, 2016.

[50] J. Li, W. Monroe, A. Ritter, M. Galley, J. Gao, and D. Jurafsky, "Deep Reinforcement Learning for Dialogue Generation," in *EMNLP*, 2016.

[51] B. Grosz, "What question would turing pose today?" *AI Magazine*, 2012.

[52] I. V. Serban, A. Sordoni, Y. Bengio, A. Courville, and J. Pineau, "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models," in *AAAI*, 2016.

[53] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollr, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in *ECCV*, 2014.

[54] A. Karpathy, *NeuralTalk2*, github.com/karpathy/neuraltalk2, 2016.

[55] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *CVPR*, 2015.

[56] AWS, *Amazon*, aws.amazon.com/ec2/, 2017.

[57] RabbitMQ, *RabbitMQ*, rabbitmq.com, 2017.

[58] G. Little, *How many turkers are there.(dec 2009)*, 2009.

[59] A. Chandrasekaran, D. Yadav, P. Chattopadhyay, V. Prabhu, and D. Parikh, "It Takes Two to Tango: Towards Theory of AI's Mind," *ArXiv preprint arXiv:1704.00717*, 2017.

[60] M. Malinowski and M. Fritz, "A multi-world approach to question answering about real-world scenes based on uncertain input," in *Advances in Neural Information Processing Systems*, 2014, pp. 1682–1690.

[61] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C Lawrence Zitnick, and D. Parikh, "Vqa: Visual question answering," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2425–2433.

[62] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, Springer, 2014, pp. 818–833.

[63] Y. Goyal, A. Mohapatra, D. Parikh, and D. Batra, "Towards transparent ai systems: Interpreting visual question answering models," *ArXiv preprint arXiv:1608.08974*, 2016.

[64] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, 2016, pp. 1135–1144.

[65] L. A. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell, "Generating visual explanations," in *European Conference on Computer Vision*, Springer, 2016, pp. 3–19.

[66] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *ArXiv preprint arXiv:1409.0473*, 2014.

[67] K. Xu, J. Ba, R. Kiros, K. Cho, A. C. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention.," in *ICML*, vol. 14, 2015, pp. 77–81.

[68] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[69] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2921–2929.

[70] D. Cosley, S. K. Lam, I. Albert, J. A. Konstan, and J. Riedl, "Is seeing believing?: How recommender system interfaces affect users' opinions," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, 2003, pp. 585–592.

[71] M. Narayanan, E. Chen, J. He, B. Kim, S. Gershman, and F. Doshi-Velez, "How do humans understand explanations from machine learning systems? an evaluation of the human-interpretability of explanation," *ArXiv preprint arXiv:1802.00682*, 2018.

[72] T. Kulesza, S. Stumpf, M. Burnett, and I. Kwan, "Tell me more?: The effects of mental model soundness on personalizing an intelligent agent," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 2012, pp. 1–10.

[73] A. Bansal, A. Farhadi, and D. Parikh, "Towards transparent systems: Semantic characterization of failure modes," in *European Conference on Computer Vision*, Springer, 2014, pp. 366–381.

[74] P. Zhang, J. Wang, A. Farhadi, M. Hebert, and D. Parikh, "Predicting failures of vision systems," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3566–3573.

[75] A. D. Dragan, K. C. Lee, and S. S. Srinivasa, "Legibility and predictability of robot motion," in *Human-Robot Interaction (HRI), 2013 8th ACM/IEEE International Conference on*, IEEE, 2013, pp. 301–308.

[76] S. I. Wang, P. Liang, and C. D. Manning, "Learning language games through interaction," in *ACL*, 2016.

[77] H. R. Pelikan and M. Broth, "Why that nao?: How humans adapt to a conventional humanoid robot in taking turns-at-talk," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ACM, 2016, pp. 4921–4932.

[78] J. Lu, J. Yang, D. Batra, and D. Parikh, "Hierarchical question-image co-attention for visual question answering," in *Advances In Neural Information Processing Systems*, 2016, pp. 289–297.

[79] A. Agrawal, D. Batra, and D. Parikh, "Analyzing the behavior of visual question answering models," *ArXiv preprint arXiv:1606.07356*, 2016.

[80] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh, "Making the v in vqa matter: Elevating the role of image understanding in visual question answering," *ArXiv preprint arXiv:1612.00837*, 2016.

[81] A. Ray, G. Christie, M. Bansal, D. Batra, and D. Parikh, "Question relevance in vqa: Identifying non-visual and false-premise questions," *ArXiv preprint arXiv:1606.06622*, 2016.

[82] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.